Amendments to the Specification:

Amend the paragraph beginning at column 1, line 5, as follows:

CROSS REFERENCE TO RELATED APPLICATIONS

This application is a continuation of U.S. Patent Application Serial No. 09/312,036 filed May 14, 1999 which is a continuation of U.S. Patent Application Serial No. 08/682,904 filed July 16, 1996, now U.S. Patent 5,959,574 which is a continuation-in-part of U.S. Patent Application Serial No. 08/404,024, filed March 14, 1995 and issued July 16, 1996 as U.S. Patent No. 5,537,119, which is a continuation-in-part of U.S. Patent Application Serial No. 08/171,327 filed December 21, 1993, now U.S. Patent No. 5,406,289.

Delete the paragraph heading and paragraph beginning at column 1, line 14.

Insert the following heading before the paragraph beginning at column 1, line 21:

a. Field of the Invention

Insert the following heading before the paragraph beginning at column 1, line 23:

b. Description of the Background

Amend the paragraph beginning at column 1, line 23, as follows:

There are many situations where the courses of multiple objects in a region must be tracked. Typically, radar is used to scan the region and generate discrete images or "snapshots" based on sets of returns or observations. In some types of tracking systems, all the returns from any one object are represented in an image as a single point unrelated to the shape or size of the objects. "Tracking" is the process of identifying a sequence of points from a respective sequence of the images that represents the motion of an object. The tracking problem is difficult when there are multiple closely spaced objects because the objects can

change[their] speed and direction rapidly and move into and out of the line of sight for other objects. The problem is exacerbated because each set of returns may result from noise as well as echoes from the actual objects. The returns resulting from the noise are also called false positives. Likewise, the radar will not detect all echoes from the actual objects and this phenomena is called a false negative or "missed detect" error. For tracking airborne objects, a large distance between the radar and the objects diminishes the signal to noise ratio so the number of false positives and false negatives can be high. For robotic applications, the power of the radar is low and as a result, the signal to noise ratio can also be low and the number of false positives and false negatives high.

Amend the paragraph beginning at column 1, line 55, as follows:

While identifying the track of an object, a kinematic model may be generated describing the[object's] location, velocity and acceleration [may be generated]of the object. Such a model provides the means by which the [object's]future motion of the object can be predicted. Based upon such a prediction, appropriate action may be initiated. For example, in a military application there is a need to track multiple enemy aircraft or missiles in a region to predict their objective, plan responses and intercept them. Alternatively, in a commercial air traffic control application there is a need to track multiple commercial aircraft around an airport to predict their future courses and avoid collision. Further,[in] these and other applications, such as robotic applications, may use radar, sonar, infrared or other object detecting radiation bandwidths for tracking objects. In particular, in robotic applications reflected radiation can be used to track a single object which moves relative to the robot (or vice versa) so the robot can work on the object.

Amend the paragraph beginning at column 2, line 5, as follows:

Consider the very simple example of two objects being tracked and no false positives or false negatives. The radar, after scanning at time t_1, reports objects at two locations in a first observation set. That is, it returns a set of two observations $\{o_{11}, o_{12}\}$. At time t_2 it returns a similar set of two observations $\{o_{21}, o_{22})$ from a second observation set. Suppose from prior processing that track data for two tracks T_1 and T_2 includes the locations at t_0 of two objects. Track T_1 may be extended through the points in the two sets of observations in any of four ways, as may track T_2. The possible extensions of T_1 can be described as: $\{T_1, o_{11}, o_{21}\}$, $\{T_1, o_{11}, o_{22}\}$, $\{T_1, o_{12}, o_{21}\}$ and $\{T_1, o_{12}, o_{22}\}$. Tracks can likewise be extended from T_2 in four possible ways, including[,] $\{T_2, o_{12}, o_{21}\}$. Figure 1 illustrates these five (out of eight) possible tracks (to simplify the problem for purposes of explanation). The five track extensions are labeled h_{11}, h_{12}, h_{13}, h_{14}, and h_{21} wherein h_{11} is derived from $\{T_1, o_{11}, o_{21}\}$, h_{12} is derived from $\{T_1, o_{11}, o_{22}\}$, h_{13} is derived from $\{T_1, o_{12}, o_{21}\}$, h_{14} is derived from $\{T_1, o_{12}, o_{22}\}$, and h_{21} is derived from $\{T_2, o_{11}, o_{21}\}$. The problem of determining which such track extensions are the most likely or optimal is hereinafter known as the assignment problem.

Amend the paragraph beginning at column 2, line 39, as follows:

Figure 2 illustrates a two by two by two matrix, c, that contains the costs for each point in relation to each possible track. The cost matrix is indexed along one axis by the track number, along another axis by the image number and along the third axis by a point number. Thus, each position in the cost

matrix lists the cost for a unique combination of points and a track, one point from each image. Figure 2 also illustrates a {0, 1} assignment matrix, z, which is defined with the same dimensions as the cost matrix. Setting a position in the assignment matrix to "one" means that the equivalent position in the cost matrix is selected into the solution. The illustrated solution matrix selects the {h_{14}, h_{21}} solution previously described. Note that for the above example of two tracks and two snapshots, the resulting cost and assignment matrices are [three dimensional] <u>three-dimensional</u>. As used in this patent application, the term "dimension" means the number of axes in the cost or assignment matrix while size refers to the number of elements along a typical axis. The costs and assignments have been grouped in matrices to facilitate computation.

Amend the paragraph beginning at column 3, line 12, as follows:

For a [3-dimensional]<u>three-dimensional</u> problem, as is illustrated in Figure 1, but with N_1 (initial) tracks, N_2 observations in scan 1, N_3 observations in scan 2, false positives and negatives assumed, the assignment problem can be formulated as:

<u>(0.1)</u>[[1.0]]

(b) *Subject to:* $$\sum_{i_2=1}^{N_2} \sum_{i_3=1}^{N_3} z_{i_1 i_2 i_3} = 1, \quad i_1 = 1, \ldots N_1\underline{,}$$

(c) $$\sum_{i_1=1}^{N_1} \sum_{i_3=1}^{N_3} z_{i_1 i_2 i_3} \leq 1, \quad i_2 = 1, \ldots N_2\underline{,}$$

(d) $$\sum_{i_1=1}^{N_1} \sum_{i_2=1}^{N_2} z_{i_1 i_2 i_3} \leq 1, \quad i_3 = 1, \ldots N_3\underline{,}$$

where "c" is the cost and "z" is a point or observation

assignment, as in Figure 2.

Amend the paragraph beginning at column 3, line 35, as follows:

The minimization equation or equivalently objective function [[1.0](a)]<u>(0.1 (a))</u> specifies the sum of the element by element product of the c and z matrices. The summation includes hypothesis representations $z_{i_1 i_2 i_3}$ with observation number zero being the gap filler observation. Equation [[1.0](b)]<u>(0.1 (b))</u> requires that each of the tracks $T_1, \ldots, T_{N_1}$ be extended by one and only one hypothesis. Equation [[1.0](c)]<u>(0.1 (c))</u> relates to each point or observation in the first observation set and requires that each such observation, except the gap filler, can only associate with one track but<u>,</u> because of the "less than" condition<u>,</u> it might not associate with any track. Equation [[1.0](d)]<u>(0.1 (d))</u> is like [[1.0](c)]<u>(0.1 (c))</u> except that it is applicable to the second observation set. Equation [[1.0](e)]<u>(0.1 (e))</u> requires that elements of the solution matrix z be limited to the zero and one values.

Amend the paragraph beginning at column 3, line 50, as follows:

The only known method to solve Problem Formulation [[0.1]]<u>(1.0)</u> exactly is a method called "Branch and Bound." This method provides a systematic ordering of the potential solutions so that solutions with a same partial solution are accessible via a branch of a tree describing all possible solutions whereby the cost of unexamined solutions on a branch are incrementally developed as the cost for other solutions on the branch are determined. When the developing cost grows to exceed the previously known minimal cost (i.e., the bound) then enumeration of the tree branch terminates. Evaluation continues with a new branch. If evaluation of the cost of a particular branch

completes, then that branch has lower cost than the previous bound so the new cost replaces the old bound. When all possible branches are evaluated or eliminated then the branch that had resulted in the last used bound is the solution. If we assume that $N_1 = N_2 = N_3 = n$ and that branches typically evaluate to half there full length, then workload associated with "branch and bound" is proportional to $(n!|\frac{n}{2}]!)^2$. This workload is unsuited to real time evaluation.

Amend the paragraph beginning at column 4, line 13, as follows:

Messrs. Frieze and Yadagar in dealing with a problem related to scheduling combinations of resources, as in job, worker and work site, showed that Problem Formulation [[0.1]](1.0) applied. They further described a solution method based upon an extension of the Lagrangian Relaxation previously mentioned. The two critical extensions provided by Messrs. Frieze and Yadagar were: (1) an iterative procedure that permitted the lower bound on the solution to be improved (by "hill climbing" described below) and (2) the recognition that when the lower bound of the relaxed problem was maximized, then there existed a method to recover the solution of the non-relaxed problem in most cases using parameters determined at the maximum. The procedures attributed to Messrs. Frieze and Yadagar are only applicable to the [3-dimensional]three-dimensional problem posed by Problem Formulation [[0.1]](1.0) and where the cost matrix is fully populated. However, tracking multiple airborne objects usually requires solution of a much higher dimensional problem.

Amend the paragraph beginning at column 4, line 31, as follows:

Figures 1 and 2 illustrate an example where "look ahead" data from the second image improved the assignment accuracy for the first image. Without the look ahead, and based only upon a

simple nearest neighbor approach, the assignments in the first set would have been reversed. Problem Formulation [[0.1]]<u>(1.0)</u> and the prior art only permit looking ahead one image. In the prior art it was known that the accuracy of assignments will improve if the process looks further ahead, however no practical method to optimally incorporate look ahead data existed. Many real radar tracking problems involve hundreds of tracks, thousands of observations per observation set and matrices with dimensions in the range of 3 to 25 including many images of look ahead.

Amend the paragraph beginning at column 5, line 14, as follows:

A previously known Multiple Hypothesis Testing (MHT) algorithm (see <u>Chapter 10 of S.S.</u> Blackman[,]<u>.</u> Multiple-Target Tracking with Radar Applications[, Chapter 10,]<u>.</u> Artech House<u>,</u> Norwood<u>,</u> MA, 1986<u>.</u>) related to formulation and scoring. The MHT procedure describes how to formulate the sparse set of all reasonable extension hypothesis (for Figure 1 the set $\{h_{11}...h_{24}\}$) and how to calculate a cost of the hypothesis $\{T_i, o_{1j}, o_{2k}\}$ based upon the previously calculated cost for hypothesis $\{T_i, o_{1j}\}$. The experience with the MHT algorithm, known in the prior art, is the basis for the assertion that look ahead through k sets of observations results in improved assignment of observations from the first set to the track.

Amend the paragraph beginning at column 5, line 26, as follows:

In theory, the MHT procedure uses the extendable costing procedure to defer assignment decision until the accumulated evidence supporting the assignment becomes overwhelming. When it makes the assignment decision<u>,</u> it then eliminates all potential assignments invalidated by the decision in a process called "pruning the tree." In practice, the MHT hypothesis tree is

limited to a fixed number of generations and the overwhelming evidence rule is replaced by a most likely and feasible rule. This rule considers each track independently of others and is therefore a local decision rule.

Amend the paragraph beginning at column 6, line 34, as follows:

In one embodiment of the invention, each k-dimensional assignment problem [2<k≤M,] $(2 < k \leq M)$ is iteratively reduced to a [k-1 dimensional] $(k-1)$-dimensional problem until a [2-dimensional] two-dimensional problem is specified or formulated. Subsequently, the [2-dimensional] two-dimensional problem formulated is solved directly and a "recovery" technique is used to iteratively recover an optimal or near-optimal solution to each k-dimensional problem from a derived [$(k$-1) dimensional] $(k-1)$-dimensional problem k= 2,3,4,...M.

Amend the paragraph beginning at column 6, line 42, as follows:

In performing each recovery step (of obtaining a solution to a k-dimensional problem using a solution to a $(k$-1)-dimensional problem), an auxiliary function[,] is utilized. In particular, to recover an optimal or near-optimal solution to a k-dimensional problem, an auxiliary function, Ψ_{k-1}[,] $(k$=4,5,...,$M)$, is specified and a region or domain is determined wherein this function is maximized, whereby values of the region determine the penalty factors of the $(k$-1)-dimensional problem such that another [2-dimensional] two-dimensional problem can be formulated which determines a solution to the k-dimensional problem using the penalty factors of the $(k$-1)-dimensional problem.

Amend the paragraph beginning at column 6, line 54, as follows:

Each Auxiliary function Ψ_k, k=3,...,M-1, is a function of both lower dimensional [problem] problems, penalty factors, and a solution at the dimension k at which the penalized cost function is solved directly (typically a [2-dimensional]two-dimensional problem). Further, in determining, for auxiliary function Ψ_k, a peak region, a gradient of the auxiliary function is determined, and utilized to identify the peak region. Thus, gradients are used for each of the approximation functions Ψ_3, Ψ_4, ..., Ψ_{M-1} which are sequentially formulated and used in determining penalty factors [penalty factors]until [tm-1]M-1 is used in determining the penalty factors for the [M-1 dimensional](M-1)-dimensional problem. Subsequently, once the M-dimensional problem is solved (using a [2-dimensional]two-dimensional problem to go from an (M-1)[dimensional]-dimensional solution to an M-dimensional solution), one or more of the following actions are taken based on the track assignments: sending a warning to aircraft or a ground or sea facility, controlling air traffic, controlling anti-aircraft or anti-missile equipment, taking evasive action, working on one of the objects.

Amend the paragraph beginning at column 7, line 3, as follows:

According to one feature of this first embodiment of the present invention, the following steps are also performed before the step of defining the auxiliary function. A preliminary auxiliary function is defined for each of the lower dimensional problems having a dimension equal or one greater than the dimension at which the penalized cost function is solved directly. The preliminary auxiliary function is a function of

lower order penalty values and a solution at the dimension at which the penalized cost function was solved directly. In determining a gradient of the preliminary auxiliary function, step in the direction of the gradient to identify a peak region of the preliminary auxiliary function and determine penalty factors at the peak region. Iteratively repeat the defining, gradient determining, stepping and peak determining steps to define auxiliary functions at successively higher dimensions until the auxiliary function [at 6-18 (k-1)]dimension (*k*-1) is determined. In an alternative second embodiment of the present invention, instead of reducing the dimentionality of the *M*-dimensional assignment problem by a single dimension at a time, a plurality of dimensions are relaxed simultaneously. This new strategy has the advantage that when the *M*-dimensional problem is relaxed directly to a [2-dimensional]two-dimensional assignment problem, then all computations may be performed precisely without utilizing an auxiliary function such as Ψ_k as in the first embodiment. More particularly, the second embodiment solves the first optimization problem (i.e., the M- dimensional assignment problem) by specifying (i.e., creating, generating, formulating and/or defining) a second optimization problem. The second optimization problem includes a second objective function and a second collection of constraint sets wherein:

a) the second objective function is a combination of the first objective function and penalty factors or terms determined for incorporating [*M-m constraint*](*M-m*)-constraint sets of the first optimization problem into the second objective function;

b) the constraint sets of the second collection include only m of the constraint sets of the first collection of constraints,

wherein [2≤m≤M-1]$2 \le m \le M-1$. Note that, once the second

optimization problem has been specified or formulated, an optimal or near-optimal solution is determined and that solution is used in specifying (i.e, creating, generating, formulating and/or defining) a third optimization problem of [*M-m*] <u>*(M-m)*</u> dimensions (or equivalently constraint sets). The third optimization problem is subsequently solved by decomposing it using the same procedure of this second embodiment as was used to decompose the first optimization problem above. Thus, a plurality of instantiations of the third optimization problem are specified, each successive instantiation having a lower number of dimensions, until an instance of the third optimization problem is a [two dimensional] <u>two-dimensional</u> assignment problem which can be solved directly. Subsequently, whenever an instance of the third optimization problem is solved, the solution is used to recover a solution to the instance of the first optimization problem from which this instance of the third optimization was derived. Thus, an optimal or near-optimal solution to the original first optimization problem may be recovered through iteration of the above steps.

Amend the paragraph beginning at column 7, line 64, as follows:

As mentioned above, the second embodiment of the present invention is especially advantageous when [m=2]<u>*m* =2</u>, since in this case all computations may be performed precisely and without utilizing auxiliary functions.

Amend the paragraph beginning at column 8, line 13, as follows:

Figure 4 is a flow chart of a process according to the prior art for solving a [3-dimensional]<u>three-dimensional</u> assignment problem.

Amend the paragraph beginning at column 8, line 15, as follows:

Figure 5 [Is]is a flow chart of a process according to the present invention for solving a k-dimensional assignment problem where "k" is greater than or equal to 3.

Amend the paragraph beginning at column 8, line 20, as follows:

Figure 7 is another block diagram of the present invention for solving the k-dimensional assignment problem where "k" is greater than or equal to [3] three (3).

Amend the paragraph beginning at column 8, line 23, as follows:

Figure 8 is a flowchart describing the procedure for solving [a] an n-dimensional assignment problem according to the second embodiment of the invention.

Amend the paragraph beginning at column 8, line 60, as follows:

The invention generates k-dimensional matrices where k is the number of images or sets of observation data in the look ahead window plus one. Then, the invention formulates a k-dimensional assignment problem as in [[1.0]](0.1) above.

Amend the paragraph beginning at column 9, line 7 as follows:

Consider the integer programming problem

$$\begin{array}{ll} \textit{Minimize} & v(z) = c^T z \\ \textit{Subject to:} & Az \le b_L \\ & Bz \le d_L \\ & z_i \textit{ is an integer for } i \in I_L \end{array} \qquad [[1.0.1]](0.2)$$

where the partitioning of the constraints is natural in some

sense. Given a multiplier vector [u≥0]$\underline{u \geq 0}$, the Lagrangian relaxation of [[1.0.1]](0.2) relative to the constraints [Bz≤d]$\underline{Bz \leq d}$ is defined to be

[ΦΦTT]

$$\underline{\Phi(u) = \text{Minimize}} \qquad \underline{\{c^T z + u^T (Bz-d)\}} \qquad \underline{(0.3)}$$

$$\text{Subject to:} \qquad [Az \leq b]\ \underline{Az \leq b,}$$
$$\underline{z_i \text{ is an integer for } i \in I.}$$

[[1.0.2]]

If the constraint set [Bz≤d]$\underline{Bz \leq d}$ is replaced by [Bz=d]$\underline{Bz = d}$, the [nonnegativity]<u>non-negativity</u> constraint on u is removed. $\mathcal{L}=c^T z+u^T(Bz-d)$ is the Lagrangian relative to the constraints [Bz≤d]$\underline{Bz \leq d}$, and hence the name Lagrangian relaxation. The following fact gives the relationship between the objective functions of the original and relaxed problems.

Amend the paragraph beginning at column 9, line 28, as follows:

FACT A.1. *If $\overline{z}$ is an optimal solution to [[1.01]]<u>(0.2)</u>, then [$\Phi \leq v(\overline{z})$] $\underline{\Phi(u) \leq v(\overline{z})}$ for all [u≥0]$\underline{u \geq 0}$. If an optimal solution $\hat{z}$ of [[1.02]]<u>(0.3)</u> is feasible for [[1.02]]<u>(0.3)</u>, then $\hat{z}$ is an optimal solution for [[1.01]]<u>(0.2)</u> and [ΦAlgorithm] $\underline{\Phi(u) = v(\overline{z})}$.*

[k $^{\infty}_{k}$=0]

Amend the paragraph beginning at column 9, line 36, as follows:

<u>This leads to the following algorithm:</u>

<u>Algorithm. Construct a sequence of multipliers</u> $\underline{\{u_k\}_{k=0}^{\infty}}$ converging to the solution $\overline{u}$ of Maximize [{Φ≥0}] $\underline{\{\Phi(u) : u \geq 0\}}$ and a corresponding sequence of feasible solutions $\{z_k\}_{k=0}^{\infty}$ of [[1.01]]<u>(0.2)</u> as follows:

1. Generate [in]<u>an</u> initial approximation u_0.

2. Given u_k, choose a search direction s_k and a search distance α_k [Ψ_k α_k $_k$ Ψ_k] so that $\Phi(u_k + \alpha_k s_k) > \Phi(u_k)$. Update the multiplier estimate u_k by [$u_{k+1}=u_k+\alpha_k$ $_k$] $u_{k+1}=u_k+\alpha_k s_k$.

3. Given u_{k+1} and a feasible solution $\hat{z}_{k+1}(u_{k+1})$ of [[1.02]](0.3), recover a feasible solution $z_{k+1}(u_{k+1})$ of the integer programming problem [[1.01]](0.2) .

4. Check the termination criteria. If the algorithm is not finished, set k=k+1 and return to Step 2. Otherwise, terminate the algorithm.

Amend the paragraph beginning at column 9, line 50, as follows:

If $\bar{z}$ is an optimal solution of [[1.01]](0.2), then

[$\Psi_k \leq \Psi \leq v(\bar{z}) \leq (z_k)$]

$\Phi(u_k) \leq \Phi(\bar{u}) \leq v(\bar{z}) \leq v(z_k)$.

Since the optimal solution [$\hat{z}=\hat{z}(u)$] $\hat{z} = \hat{z}(u)$ of [[1.02]](0.3) is usually not a feasible solution of [[1.01]](0.2) for any choice of the multipliers,

[Φ

Φ α α

Φ

α α]

$\Phi(\bar{u})$ is usually strictly less than $v(\bar{z})$. The difference $v(\bar{z})-\Phi(\bar{u})$ is called the "duality gap," which can be estimated by comparing the best relaxed and recovered solutions computed in the course of maximizing $\Phi(u)$.

Amend the paragraph beginning at column 9, line 64 as follows:

[$_p \leq p \leq k$]

Thus, to eliminate the "<" in the constraint formulations (e.g., (0.1)(a) - (0.1)(d)) that resulted from false positives, the constraint sets have been modified, as indicated above, to

incorporate a gap filler in each observation set to account for false positives. Thus, a zero for an index, i_p, $1 \le p \le k$ in $z^k_{i_1 \ldots i_k}$ in problem [[1.1]](0.4) below indicates that the track extension represented by $z^k_{i_1 \ldots i_k}$ includes a false positive in the p^{th} observation set. Note that this implies that a hypothesis be formed incorporating an observation with *k*-1 gap fillers, e.g., $z_{0 \ldots 0 i_p 0 \ldots 0}$, $i_p \neq 0$. Thus, the resulting generalization of Problem Formulation [[1.0]](0.1) without the "less than" complication within the constraints is the following *k*-Dimensional Assignment Problems in which [k≥3]$k \ge 3$:

[[1.1]] (0.4)

$$\text{Minimize:} \quad \sum_{i_1=0}^{N_1} \cdots \sum_{i_k=0}^{N_k} c_{i_1 \ldots i_k} z_{i_1 \ldots i_k}$$

$$\text{Subject to:} \quad \sum_{i_2=0}^{N_2} \cdots \sum_{i_k=0}^{N_k} z_{i_1 \ldots i_k} = 1, \qquad i_1 = 1, \ldots N_1,$$

$$\sum_{i_1=0}^{N_1} \cdots \sum_{i_{j-1}=0}^{N_{j-1}} \sum_{i_{j+1}=0}^{N_{j+1}} \cdots \sum_{i_k=0}^{N_k} z_{i_1 \ldots i_k} = 1,$$

$$\text{for } i_j = 1, \ldots N_j, \text{ and } j = 2, \ldots k-1,$$

$$\sum_{i_1=0}^{N_1} \cdots \sum_{i_{k-1}=0}^{N_{k-1}} z_{i_1 \ldots i_k} = 1, \qquad i_k = 1, \ldots N_k,$$

$$z_{i_1 \ldots i_k} \in \{0,1\}, \qquad \forall i_n \; n = 1, \ldots, k,$$

where c and z are similarly dimensioned matrices representing costs and hypothetical assignments. Note, in general, for tracking problems these matrices are sparse.

Amend the paragraph beginning at column 10, line 29, as follows:

After formulating the k-dimensional assignment problem as in [[1.1]] (0.4), the present invention solves the resulting problem so as to generate the outputs required by devices 110, 112, 122 and 130. For each observation set O_i received from converter 104 at time t_i where $i=1,\ldots,N_1$, the computer processes O_i in a batch together with the other observation sets $O_{i-k+1},\ldots,O_i$ and the track T_{i-k}, i.e., T_j is the set of all tracks that have been defined up to but not including O_j. (Note, bold type designations refer to the vector of elements for the indicated time, i.e., the set of all observations in the scan or tracks existing at the time, etc.) The result of this processing is the new set of tracks T_{i-k+1} and a set of cost weighted possible solutions indicating how the tracks might extend to the current time t_i. At time t_{i+1} the batch process is repeated using the newest observation set and deleting the oldest. Thus, there is a moving window of observation sets which is shifted forward to always incorporate the most recent observation set. The effect is that input observation sets are reused for [k-1] (k-1) cycles and then on the observation set's [k-th] k^{th} reuse each observation within the observation set is integrated into a track.

Amend the paragraph beginning at column 10, line 50, as follows:

Figure 7 illustrates various processes implemented upon receipt of each observation set. Except for the addition of the k-dimensional assignment solving process 300 and the modification to scoring process 154 to build data structures suitable for process 300, all processes in Figure 7 are based upon prior art. The following processes 150 and 152 extend previously defined

tracks $h_{i-1[.]}$ based on new observations. Gate formulation and output process 156 determines, for each of the previously defined tracks, a zone wherein the track may potentially extend based on limits of velocity, maneuverability and radar precision. One such technique to accomplish this is the cone method described previously. The definition of the zone is passed to gating process 150. When a new observation set O_i is received, the gating process 150 will match each member observation with the zone for each member of the hypothetical set h_{i-1}. After all input observations from O_i are processed, the new hypothesis set h_i is generated by extending each track of the prior set of hypothetical tracks h_{i-1} either with missed detect gap fillers or with all new observation elements satisfying the track's zone. This is a [many to many] many-to-many matching in that each hypothesis member can be extended to many new observations and each new observation can be used to extend many hypotheses. It, however, is not a full matching in that any hypothesis will neither be matched to all observations nor vice versa. It is this matching characteristic that leads to the sparse matrices involved in the tracking process. Subsequently, gating 150 forwards the new hypothesis set h_i to filtering process 152. Filtering process 152 determines a smooth curve for each member of h_i. Such a smooth curve is more likely than a sharp turn from each point straight to the next point. Further, the filtering process 152 removes small errors that may occur in generating observations. Note that in performing these tasks, the filtering process 152 preferably utilizes a minimization of a least squares test of the points in a track hypothesis or a Kalman [Filtcring] filtering approach.

Amend the paragraph beginning at column 11, line 20, as follows:

As noted above, the foregoing track extension process requires knowledge of a previous track. For the initial observations, the following gating process 158 and filtering process 160 determine the "previous track" based on initial observations. In determining the initial tracks, the points from the first observation set form the beginning points of all possible tracks. After observation data from the next observation set is received, sets of simple [two point] <u>two-point</u> straight line tracks are defined. Then, promotion, gate formulation, and output step 162 determines a zone in which future extensions are possible. Note that filtering step 160 uses curve fitting techniques to smooth the track extensions depending upon the number of prior observations that have accumulated in each hypothesis. Further note that promotion, gate formulation and output process 162 also determines when sufficient observations have accumulated to form a basis for promoting the track to processes 150 and 152 as described above.

Amend the paragraph beginning at column 12, line 3, as follows:

The assignment solving process 300 is described below. Its output is simply the list of assignments which constitute the most likely solution of the problem described by Equation [[1.1]<u>(0.4)</u>. Note that both gate formulation and output processes 156 and 162 use (at different times) the list of assignments to generate the updated track history T_i to eliminate or prune alternative previous hypotheses that are prohibited by the actual assignments in the list, and<u>,</u> subsequently<u>,</u> to output any required data. Also note that when one of the gate formulation and output processes 156 and 162 accomplish these tasks, the process will subsequently generate and forward the new

set of gates for each remaining hypothesis and the processes will then be prepared to receive the next set of observations. In one embodiment, the loop described here will generate zones for a delayed set of observations rather than the subsequent set. This permits processes 156 and 162 to operate on even observation sets while the scoring step 154 and k-dimensional solving process operate on odd sets of observations, or vice versa.

Amend the paragraph beginning at column 12, line 22, as follows:

The assignment solving process 300 permits the present invention to operate with a window size of dimension k-1 for [k≥3]some $k \geq 3$. The upper limit on k depends only upon the computational power of the computer 106 and the response time constraints of system 100. The k-1 observation sets within the processing window plus the prior track history result in a k-dimensional Assignment Problem as described by Problem Formulation [[1.1]](0.4). The present invention solves this generalized problem including the processes required to consider false positives and negatives, and also the processes required to consider sparse matrix problem formulations.

Amend the paragraph beginning at column 12, line 35, as follows:

In describing a first embodiment of the k-dimensional assignment solver 300, it is worthwhile to also discuss the process of Figure 4 which is used by the solver 300. Figure 4 illustrates use of the Frieze and Yadagar process as shown in prior art for transforming a [3-dimensional] three-dimensional assignment problem into a [2-dimensional] two-dimensional assignment problem and then use a hill climbing algorithm to solve the [3-dimensional] three-dimensional assignment problem. The solver 300 uses a Lagrangian Relaxation technique (well known

in the art) to reduce the dimension of an original [k-dimensional] k-dimensional assignment problem ([k>3] $k > 3$) down to a [3-dimensional] three-dimensional problem and then use the process of Figure 4 to solve the [3-dimensional] three-dimensional problem. Further note that the Lagrangian Relaxation technique is also utilized by the process of Figure 4 and that in using this technique the requirement that each point is assigned to one and only one track is relaxed. Instead, an additional cost, which is equal to a respective Lagrangian Coefficient ["u"] u, is added to the cost or objective function [[1.0 (a)]] (0.1) (a) whenever a point is assigned to more than one track. This additional cost can be picked to weight the significance of each constraint violation differently, so this additional cost is represented as a vector of coefficients u which are correlated with respective observation points. Hill climbing will then develop a sequence of Lagrangian Coefficients sets designated $(u_0,\ldots,u_j,\ u_{j+1},\ldots,u_p)$. That correspond to an optimum solution of the [2-dimensional] two-dimensional assignment problem. The assignments at this optimum solution are then used to "recover" the assignment solution of the [3-dimensional] three-dimensional assignment problem.

Amend the paragraph beginning at column 12, line 63, as follows:

In step 200 of Figure 4, initial values are selected for the u_0 coefficients. Because the Lagrangian Relaxation process is iterative, the initial values are not critical and are all initially selected as zero. In step 202, these additional costs are applied to the objective function [[1.0 (a)]] (0.1 (a). With the addition of the costs ["u"] u, the goal is still to assign the points which minimize the total cost. This transforms Equation [[1.0 (a)]] (0.1 (a), written for [k=3] $k = 3$ and altered to exclude mechanisms related to false positives and negatives,

into objective function [[2.1 (a)]] (1.1) (a). In the first iteration it is not necessary to consider the ["u"] u matrix because all ["u"] u values are set to zero. To relax the requirement that each point be assigned to one and only one track, the constraint Equation [[1.0 (d)]] (0.1 (d) is deleted, thereby permitting points from the last image to be assigned to more than one track. Note that while any axis can be chosen for relaxation, observation constraints are preferably relaxed. The effect of this relaxation is to create a new problem which must have the same solution in the first two axes but which can have a [differing] different solution in the third axis. The result is constraints [[2.1 (b-d)]] (1.1) (b-d).

Amend the paragraph beginning at column 13, line 17, as follows:

[[2.1]] (1.1)

(a) Minimize[:] $$\sum_{i_1=0}^{N_1} \sum_{i_2=0}^{N_2} \{(c_{i_1 i_2 i_3} - u[_j]_{i_3})\} z_{i_1 i_2 i_3}$$

(b) Subject to: $$\sum_{i_2=1}^{N_2} \sum_{i_3=1}^{N_3} z_{i_1 i_2 i_3} = 1, \qquad i_1 = 1, \ldots, N_1,$$

(c) $$\sum_{i_1=1}^{N_1} \sum_{i_3=1}^{N_3} z_{i_1 i_2 i_3} \le 1, \qquad i_2 = 1, \ldots, N_2,$$

(d) $$z_{i_1 i_2 i_3} \in \{0,1\} \qquad \forall\ z_{i_1 i_2 i_3}.$$

Amend the paragraph beginning at column 13, line 47, as follows:

Step 204 then generates from the [3-dimensional] three-dimensional problem described by Problem Formulation [[1.0]] (0.1) a new [2-dimensional] two-dimensional problem formulation which will have the same solution for the first two indices. As

Problem Formulation [[2.1]] (1.1) has no constraints on the 3rd axis, any value within a particular 3rd axis can be used in a solution, but using anything other than the minimum value from any [3-rd] 3rd axis has the effect of increasing solution cost. Conceptually, the effect of step 204 is to change the [3-dimensional] three-dimensional arrays in Problem Formulation [[2.2]] (1.1) into [2-dimensional] two-dimensional arrays as shown in Problem Formulation [[2.1]] (1.2) and to generate the new [2-dimensional] two-dimensional matrix $m_{i_1 i_2}$ defined as shown in Equation [[2.3]] (1.3).

Amend the paragraph beginning at column 13, line 33, as follows:

[[2.2]] (1.2)

(a) Minimize[:] $\sum_{i_1=0}^{N_1} \sum_{i_2=0}^{N_2} \left\{ \begin{matrix} Min: \\ \forall\ i_3 \end{matrix} (c_{i_1 i_2 i_3} - u[_j]_{i_3}) \right\} z_{i_1 i_2}$

(b) Subject to: $\sum_{i_2=1}^{N_2} \sum_{i_3=1}^{N_3} z_{i_2 i_3} = 1, \quad i_1 = 1, \ldots, N_1$

(c) $\sum_{i_1=1}^{N_1} \sum_{i_3=1}^{N_3} z_{i_1 i_3} \le 1, \quad i_2 = 1, \ldots, N_2$

(d) $z_{i_1 i_2} \in \{0,1\} \quad \forall\ z_{i_1 i_2}$

[[2.3]]] (1.3)

$$m_{i_1 i_2} = \text{Min: arg minimize} \{ c_{i_1 i_2} - u_{j_i} \mid i=1, \ldots, N_k \}$$

Amend the paragraph beginning at column 13, line 64, as follows:

The cost or objective function for the reduced problem as

defined by [[2.2 (a)]] (1.2) (a), if evaluated at all possible values of u is a surface over the domain of[U^3] u. This surface is referred to as [[Φu]] $\Phi(u)$ and is non-smooth but provable convex (i.e., it has a single peak and several other critical characteristics which form terraces). [Because of]Due to the convex characteristics of $\Phi(u)$, the results from solving Problem Formulation(1.2) at any particular u_j can be used to generate a new set of

[Φ u

u_j]

coefficients u_{j+1} whose corresponding [Problem Formulation [2.2] problem solution is a] cost value is closer to the peak of [Φ u U_P] $\Phi(u)$. The particular set of Lagrangian Coefficients that will generate the two-dimensional problem resulting in the maximum cost is designated u_p. To recover the solution to the three-dimensional [dimensional] assignment problem requires solving the Equation [2.2] (1.2) problem corresponding to u_p.

Amend the paragraph beginning at column 14, line 15, as follows:

In step 206, the [two dimensional] (two-dimensional) problem is solved directly using a technique known to those skilled in the art such as Reverse Auction for the corresponding cost and solution values. This is the evaluation of one point on the surface or for the first iteration $\Phi(u_0)$.

Amend the paragraph beginning at column 14, line 21, as follows:

Thus, after this first iteration, the points have been assigned based on all "*u*" values being arbitrarily set to zero. Because the "*u*" values have been arbitrarily assigned, it is unlikely that these assignments are correct and it is likely that further iterations are required to properly assign the points.

Step 208 determines whether the points have been properly assigned after the first iteration by determining if for this set of assignments whether a different set of "*u*" values could result in a higher total cost. Thus, step 208 is implemented by determining the gradient of objective function [[2.2] (a)] <u>(1.2 (a)</u> with respect to u_j. If the gradient is substantially non-zero (greater than a predetermined limit) then the assignments are not at or near enough to the peak of the [Φu]

[U_{j+1}]

<u>surface Φ(u) (decision 210), and the new set of Lagrangian Coefficients u_{j+1} is determined.</u>

Amend the paragraph beginning at column 14, line 36, as follows:

Hill climbing Step 212 determines the u_{j+1} values based upon the u_j values, the direction resulting from protecting the previous gradient into the U^3 domain, and a step size. The solution value of the [2-dimensional] <u>two-dimensional</u> problem is the set of coefficients that minimize the [2-dimensional] <u>two-dimensional</u> problem and the actual cost at the minimum. Those coefficients augmented by the coefficients stored in $m_{i_1 i_2}$ permit the evaluation (but not the minimization) of the cost term in Problem Formulation [[2.1]] <u>(1.1)</u>. These two cost terms are lower and upper bounds on the actual minimized cost of the [3-dimensional] <u>three-dimensional</u> problem, and the difference between them in combination with the gradient is used to compute the step size.

Amend the paragraph beginning at column 14, line 50, as follows:

With this new set of ["u"] <u>u</u> values, steps 202-210 are repeated as a second iteration. Steps 212 and 202-210 are repeated until the gradient as a function of u determined in step 208 is less than the predetermined limit. This indicates that

the u_p values which locate the peak area of the $\Phi(u)$ surface are determined and that the corresponding Problem Formulation [[2.2]] (1.2) has been solved. Step 214 will attempt to use the assignments that resulted from this particular [2-dimensional] two-dimensional assignment problem to recover the solution of the [3-dimensional] three-dimensional assignment problem as described below. If the limit was chosen properly so that the ["u"] u values are close enough to the peak, this recovery will yield the proper set of assignments that rigidly satisfies the constraint that each point be assigned to one and only one track. However, if the ["u"] u values are not close enough to the peak, then the limit value for decision 210 is reduced and the repetition of steps 212 and 202-210 is continued.

Amend the paragraph beginning at column 15, line 1, as follows:

Step 214 recovers the [3-dimensional] three-dimensional assignment solution by using the assignment values determined on the last iteration through step 208. Consider the [2-dimensional] two-dimensional z assignment matrix to have 1's in the locations specified by the list $L_1=(a_i, b_i)_{i=1}^{N}$. If the [3-dimensional] three-dimensional z matrix is specified by placing 1's at the location indicated by the list $L_2=(a_i, b_i, m_{a_i b_i})_{i=1}^{N}$ then the result is a solution of Problem Formulation [[2.1]] (1.1). Let $L_3=(m_{a_i b_i})_{i=1}^{N}$ be the list formed by the third index. If each member of L_3 is unique then the L_2 solution satisfies the third constraint so it is a solution to Problem Formulation [[1.0]] (0.1). When this is not the case, recovery determines the minimal substitutions required within list L_3 so that it plus L_1 will be a feasible solution, i.e., a solution which satisfies the constraints of a problem formulation, but which may not optimize the objective function of the problem formulation. This stage of the recovery process is formulated as a [2-dimensional] two-

dimensional assignment problem[:] as follows. Form a new cost matrix $[c_{i,j}]_{i,j=1}^{N}$ where $c_{i,j}=c_{a_i b_i,j}$ for $j=1\ldots N_i$ and the N_i term is the total number of cost elements in the selected row of the [3-dimensional] three-dimensional cost matrix. Attempt to solve this [2-dimensional] two-dimensional problem for the minimum using two constraints sets. If a feasible solution is found then the result will have the same form as list L_1. Replace the first set of [indexes]indices by the indicated (a_i, b_i) pairs taken from list L_1 and the result will be a feasible solution of Problem Formulation [[1.1]] (0.4). If no feasible solution to the new [2-dimensional] two-dimensional problem exists then further effort to locate the peak of [Φ u] $\Phi(u)$ is required.

Delete the paragraph beginning at column 15, line 33, thru line 45.

Amend the paragraph beginning at column 15, line 48, as follows:

I.1. Generalization to a Multi-Dimensional Assignment Solving Process

Let M be a fixed integer and assume that M is the dimension of the initial assignment problem to be solved. Thus, initially, the result of the scoring step 154 is a M-dimensional Cost Matrix which is structured as a sparse matrix (i.e., only a small percentage of the entries in the cost and assignment matrices are filled or non-zero). Individual cost elements represent the likelihood that a track T_i as extended by the set of observations $\{o_{ij} | i=1,\ldots,M-1\}$, is not valid. Because the matrix is sparse the list of cost elements is stored as a packed list, and then for each dimension of the matrix, a vector of a variable length list of pointers to the cost elements is generated and stored. This organization means that for a particular observation [o_{ij}]O_{ij} for the j^{th} list in the i^{th} vector will be a list of pointers to all

hypotheses in which [o_{ij}]$\underline{O_{ij}}$ participates. This structure is further explained in the following section dealing with problem partitioning.

Amend the paragraph beginning at column 15, line 58, as follows:

The objective of the assignment solving process is to select from the set of all possible combinations of track extensions a subset that satisfies two criteria. First, each point in the subset of combinations should be assigned to one and only one track and therefore, included in one and only one combination of the subset, and second, the total of the scoring sums for the combinations of the subset should be minimized. This yields the following *M*-dimensional equations where [k=M] $\underline{k = M}$:

[[3.1]] (1.4)

(a) Minimize: $v_k(z^k) = \sum_{i_1=0}^{N_1} \ldots \sum_{i_k=0}^{N_k} c^k_{i_1 \ldots i_k} z^k_{i_1 \ldots i_k}$

(b) Subject to: $\sum_{i_2=0}^{N_2} \ldots \sum_{i_k=0}^{N_k} z^k_{i_1 \ldots i_k} = 1 \qquad i_1 = 1, \ldots N_1$,

(c) $\sum_{i_1=0}^{N_1} \ldots \sum_{i_{j-1}=0}^{N_{j-1}} \sum_{i_{j+1}=0}^{N_{j+1}} \ldots \sum_{i_k=0}^{N_k} z^k_{i_1 \ldots i_k} = 1$

for $i_j = 1, \ldots N_j$
and $j = 2, \ldots k-1$,

(d) $\sum_{i_1=0}^{N_1} \ldots \sum_{i_{k-1}}^{N_{k-1}} z^k_{i_1 \ldots i_k} = 1 \qquad i_k = 1, \ldots N_k$,

(e) $z^k_{i_1 \ldots i_k} \in \{0,1\}$ for all $i_1 \ldots l_k$,

and where c^k is the cost matrix [$c^k_{i_1 \ldots i_k}$] which is a function of the distance between the observed point z^k and the smoothed track

determined by the filtering step, and v_k is the cost function. This set of equations is similar to the set presented in Problem Formulation [[1.1]] (0.4) except that it includes the subscript and superscript k notation. Thus, in solving the M-dimensional Assignment Problem the invention reduces this problem to an [M-1 dimensional] (M-1)-dimensional Assignment Problem and then to an [N-2 dimensional] (N-2)-dimensional Assignment Problem, etc. Further, the symbol $k \in \{3, \ldots, M\}$ customizes Problem Formulation [[3.1]] (0.4) to a particular relaxation level. That is, the notation is used to reference data from levels relatively removed as in c^{k+1} are the cost coefficients which existed prior to this level of relaxed coefficients c^k. Note that actual observations are numbered from 1 to N_i, where N_i is the number of observations in observation set i. Further note that the added [0] 0 observation in each set of observations is the unconstrained "gap filler." This element serves as a filler in substituting for missed detects, and a sequence of these elements including only one true observation represents the possibility that the observation is a false positive. Also note that by being unconstrained a gap filler may be used in as many hypotheses as required.

Amend the paragraph beginning at column 16, line 45, as follows:

While direct solution to [[3.1]] (0.4) would give the precise assignment, the solution of k-dimensional equations directly for large k is too complex and time consuming for practice. Thus, the present invention solves this problem indirectly.

Amend the paragraph beginning at column 16, line 50, as follows:

The following is a short description of many aspects of the

present invention and includes some steps according to the prior art. The first step in solving the problem indirectly is to reduce the complexity of the problem by the previously known and discussed Lagrangian Relaxation technique. According to the Lagrangian Relaxation technique, the absolute requirement that each point is assigned to one and only one track is relaxed such that for some one image, points can be assigned to more than one track. However, a penalty based on a respective Lagrangian Coefficient u^k is added to the cost function when a point in the image is assigned to more than one track. The Lagrangian Relaxation technique reduces the complexity or "dimension" of the formulation of the assignment problem because constraints on one observation set are relaxed. Thus, the Lagrangian Relaxation is performed iteratively to repeatedly reduce the dimension until a [2-dimensional] two-dimensional penalized cost function problem results as in Problem Formulation [[2.1]] (1.1). This [2-dimensional] two-dimensional problem is solved then directly by a previously known technique such as Reverse Auction. The penalized cost function for the [2-dimensional]two-dimensional problem defines a valley or convex shaped surface which is a function of various sets of $\{u^k|k=3,...,M\}$ penalty values and one set of assignments for the points in two dimensions. That is, for each particular u^3 there is a corresponding [2-dimensional] two-dimensional penalized cost function problem and its solution. Note that the solution of the [2-dimensional] two-dimensional penalized cost function problem identifies the set of assignments for the particular u^3 values that minimize the penalized cost function. However, these assignments are not likely to be optimum for any higher dimensional problem because they were based on an initial arbitrary set of u_k values. Therefore, the next step is to determine the optimum assignments for the related [3-dimensional] three-dimensional penalized cost function

problem. There exists a [2-dimensional] two-dimensional hill shaped function [Φ

u^3 $u^k|k>3\}$]

Φ which is a graph of the minimums of all penalized cost functions at various sets of assignments in two dimensions. For the three-dimensional problem, the function *Φ* can be defined based on the solution to the foregoing two-dimensional penalized cost function. By using the current u^3 values and the $\{u^k|k > 3\}$ values originally assigned[. Then], the gradient of the hill-shaped [Φ

u^3]

function *Φ* is determined, which points toward the peak of the hill. By using the gradient and u^3 values previously selected for the one point on the hill (corresponding to the minimum of the penalized cost [Φ

U^3]

function *Φ*) as a starting point, the u^3 values can be found for which the corresponding problem will result in the peak of the [Φ

α α]

function *Φ*. The solution of the corresponding two-dimensional problem is the proper values for two of the three sets of indices in the three-dimensional problem. These solution indices can select a subsection of the cost array which maps to a two-dimensional array. The set of indices which minimize the two-dimensional assignment problem based on that array corresponds to the proper assignment of points in the third dimension. The foregoing "recovery" process was known in the prior art, but it is modified here to adjust for the sparse matrix characteristic. The next task is to recover the solution of the proper u^4 values for the [4-dimensional] four-dimensional problem. The foregoing hill climbing process will not work again because the foregoing

hill climbing process when required to locate the [4-dimensional] four-dimensional u^4 values for the peak of [Φ^3 case of Φ^2 Φ^3] Φ^3 requires the exact definition of the function Φ^3 (as was available in the case of Φ^2) or an always less than approximation of Φ^3, whereas the iteration can result in a greater than approximation of [Φ^3

function ψ α α

Φ

U^k

gradient of the ψ

u^4 u^4]

Φ^3. According to the present invention, another three-dimensional function Ψ is defined which is a "less than approximation" the three-dimensional function Φ and which can be defined based on the solution to the two-dimensional penalized cost function and the previously assigned and determined u^k values. Next, the gradient of the function Ψ is found and hill climbing technique used to determine the u^4 values at the peak. Each selected u^4 set results in a new [3-dimensional] three-dimensional problem and requires the [2-dimensional] two-dimensional hill climbing based upon new [2-dimensional] two-dimensional problems. At the peak of the [3-dimensional Ψ] three-dimensional function Ψ, the solution values are a subset of the values required for the four-dimensional solution. Recovery processing extends the subset to a complete solution. This process is repeated iteratively until the u^k values that result in a corresponding solution at the peak of the highest order [Ψ Φ

Φ]

functions Ψ and Φ are found. The final recovery process then results in the solution *of k*-dimensional problem. Figure 5 illustrates process 300 in more detail.

Insert the following paragraph before the paragraph beginning at column 18, line 47:

I.2. Problem Formulation

In problem formulation step 310, all data structures for the subsequent steps are allocated and linked by pointers as required for execution efficiency. The incoming problem is partitioned as described in the subsequent section. This partitioning has the effect of dividing the incoming problem into a set of independent problems and thus reducing the total workload. The partitioning process depends only on the actual cost matrix so the partitioning can and is performed for all levels of the relaxation process.

Amend the paragraph beginning at column 18, line 47, as follows:

I.2.1. Relaxation and Recovery

Step 320 begins the Lagrangian Relaxation process for reducing the *M*-dimensional problem by selecting all Lagrangian Coefficient u^M penalty values initially equal to zero. The Lagrangian Coefficients associated with the M^{th} constraint set are [a N_{M+1} element] an (N_M+1)-element vector. The reduction of this *M*-dimensional problem in step 322 to a [M-1 dimensional] *(M-1)*-dimensional problem uses the two step process described above. First, a penalty based on the value of the respective u^M coefficient is added to the cost function when a point is assigned to more than one track and then the resultant cost function is minimized. However, during the first iteration, the penalty is zero because all u^M values are initially set to zero. Second, the requirement that no point from any image can be assigned to more than one track is relaxed for one of the images. In the extreme this would allow a point from the relaxed image to be associate with every track. However, the effect of the previous penalty would probably mean that such an association

would not minimize the cost. The effect of the two steps in combination is to remove a hard constraint while adding the penalty to the cost function so that it operates like a soft constraint. For step 322 this [two step] two-step process results in the following penalized cost function problem with *k=M*:

(a) $\Phi_k(u^k) \equiv$ Minimize: $\phi_k(u^k, z^k)$ [[3.2]] (1.5)

$$= \sum_{i_1=0}^{N_1} \cdots \sum_{i_k=0}^{N_k} c^k_{i_1 \ldots i_k} z^k_{i_1 \ldots i_k} - \sum_{i_k=0}^{N_k} u^k_{i_k} \left[\sum_{i_1=0}^{N_1} \cdots \sum_{i_{k-1}}^{N_{k-1}} z^k_{i_1 \ldots i_k} - 1 \right]$$

$$= \sum_{i_1=0}^{N_1} \cdots \sum_{i_k=0}^{N_k} (c^k_{i_1 \ldots i_k} - u^k_{i_k}) z^k_{i_1 \ldots i_k} + \sum_{i_k=0}^{N_k} u^k_{i_k}$$

(b) Subject to: $\sum_{i_2=0}^{N_2} \cdots \sum_{i_k=0}^{N_k} z^k_{i_1 \ldots i_k} = 1 \qquad i_1 = 1, \ldots N_1$.

(c) $\sum_{i_1=0}^{N_1} \cdots \sum_{i_{j-1}=0}^{N_{j-1}} \sum_{i_{j+1}=0}^{N_{j+1}} \cdots \sum_{i_k=0}^{N_k} z^k_{i_1 \ldots i_k} = 1$

for $i_j = 1, \ldots N_j$
and $j = 2, \ldots k-1$,

(d) $z^k_{i_1 \ldots i_k} \in \{0,1\}$ for all $i_1 \ldots l_k$.

Because the constraint on single assignment of elements from the last image has been eliminated, [a M-1 dimensional] an *(M-1)*-dimensional problem can be developed by eliminating some of the possible assignments. As shown in Equations [[3.3]] (1.6), this is done by selecting the smallest cost element from each of the M^{th} axis vectors of the cost matrix. Reduction in this manner yields a new, [lower order] lower-order penalized cost function defined by Equations [[3.3]](1.6) which has the same minimum cost as does the objective function defined by [[3.2 (a)]](1.5)(a) above.

Amend the paragraph beginning at column 19, line 41, as follows:

The cost vectors are selected as follows. Define [an index array $m_{1\ldots ik-1}$ and] a new cost array $c^{k-1}_{i_1 \ldots i_{k-1}}$ by:

[[3.3]]

$$[m^k_{i_1 \ldots i_k} = \text{Min: arg minimize } \{c^k_{i_1 \ldots i_k} - u^k_{i_k} \mid i_k = 0, 1, \ldots, N_k\}]$$

(1.6)

$$c^{k-1}_{i_1 \ldots i_{k-1}} = c^k_{i_1 \ldots i_{k-1} m^k_{i_1 \ldots i_k}} \qquad \text{for}(i_1, \ldots, i_{k-1}) \neq (0, \ldots, 0)$$

$$c^{k-1}_{0,\ldots,0} = \sum_{i_k=0}^{N_k} \min\{0, c^k_{0\ldots 0} - u^k_{i_k}\}$$

The resulting [M-1 dimensional] $(M-1)$-dimensional problem is (where $k=M$):

[[3.4]] (1.7)

$$\Phi_k(u^k) = \text{Minimize: } v_{k-1}(z^{k-1}) = \sum_{i_1=0}^{N_1} \cdots \sum_{i_{k-1}=0}^{N_{k-1}} c^{k-1}_{i_1 \ldots i_{k-1}} z^{k-1}_{i_1 \ldots i_{k-1}}$$

Subject to:

$$\sum_{i_2=0}^{N_2} \cdots \sum_{i_k=0}^{N_k} z^{k-1}_{i_1 \ldots i_k} = 1 \qquad i_1 = 1, \ldots N_1$$

$$\sum_{i_1=0}^{N_1} \cdots \sum_{i_{j-1}=0}^{N_{j-1}} \sum_{i_{j+1}=0}^{N_{j+1}} \cdots \sum_{i_k=0}^{N_k} z^{k-1}_{i_1 \ldots i_k} = 1$$

for $i_j = 1, \ldots N_j$
and $j=2, \ldots k-1$

$$\sum_{i_1=0}^{N_1} \cdots \sum_{i_{k-2}=0}^{N_{k-2}} z^{k-1}_{i_1 \ldots i_{k-1}} = 1 \qquad i_{k-1} = 1, \ldots N_{k-1}$$

$$z^{k}_{i_1 \ldots i_k} \in \{0,1\} \qquad \text{for all } i_1 \ldots l_{k-1}.$$

Assignment Problem [[3.1]] (1.4) and Problem Formulation [[3.4]] (1.7) differ only in the dimension *M* vs. *M*-1, respectively. An optimal solution to [[3.4]] (1.7) is also an optimal solution to equation [[3.2]] (1.5). This relationship is the basis for an iterative sequence of reductions indicated by steps 320-332 through 330-332 and 200-204 in which the penalized cost function problem is reduced to a [two dimensional] two-dimensional problem. As these formula will be used to describe the processing at all levels, the lowercase *k* is used except where specific reference to the top level is needed. In step 206, the [2-dimensional] two-dimensional penalized cost function is solved directly by the prior art Reverse Auction technique. Each execution of 206 produces two results, the set of z^2 values that minimize the problem and the cost that results v^2 when these z values are substituted into the objective function [[3.4 (a)]] (1.7)(a).

Amend the paragraph beginning at column 20, line 37, as follows:

In step 208, according to the prior art, solution z^2 values are substituted into the [2-dimensional] two-dimensional derivative of the [Φ_2]surface Φ_2. The result indicates how the value of u^3 should be adjusted so as to perform the hill climbing function. As was previously described the objective is to produce a sequence of u^3_i values which ends when the u^3_p value is in the domain of the peak of the [Φ]surface Φ. The section "Determining Effective Gradient" describes how new values are computed and how it is determined that the current u^k_i points

[α α

u^k]

to the peak of Φ_2. When no further adjustment is required the flow moves to step 214 which will attempt to recover the [3-dimensional] three-dimensional solution as previously described. When further adjustment is required then the flow progresses to step 212 and the new values of u^k are computed. At the [2-dimensional] two-dimensional level the method of the prior art could be used for the hill climbing procedure. However, it is not practical to use this prior art hill climbing technique to determine the updated Lagrangian Coefficients u^k or the Max on the next (or any) higher order [Φ] surface Φ because the next (or any) higher dimensional function Φ cannot be defined based on known information.

Delete the paragraph beginning at column 21, line 1, thru column 21, line 10.

Amend the paragraph beginning at column 21, line 12, as follows:

Instead, the present invention defines a new function based on known information which is useful for hill climbing from the third to the fourth and above dimensions, i.e., u^k values which result in z values that are closer to the proper z values for the highest k-dimension. This hill climbing process (which is different than that used in the prior art of Figure 4 for recovering only the [three dimensional] three-dimensional solution) is used iteratively at all lower dimensions k of the M-dimensional problem (including the [3-dimensional] three-dimensional level where it replaces prior art) even when k is much larger than three. Figure 6 helps to explain this new hill climbing technique and illustrates the original k-dimensional cost function v_k of Problem Formulation [[3.1]] (1.4). However, the actual k-dimensional cost surface v_k defined by [[3.1 (a)]] (1.4) (a) comprises scalar values at each point described by k-

dimensional vectors and as such can not be drawn. Nevertheless, for illustration purposes only, Figure 6 ignores the reality of ordering vectors and illustrates a concave function $v_k(z^k)$ to represent Equation [[3.1]] (1.4). The[v_k] surface v_k is illustrated as being smooth to simplify the explanation although actually it can be imagined to be terraced. The goal of the assignment problem is to find the values of $\overline{\mathbf{z}}^k$; these values minimize the k-dimensional cost function v_k.

Amend the paragraph beginning at column 21, line 32, as follows:

For purposes of explanation, assume that in Figure 6, $k=4$ (the procedure is used for all [$k \geq 3$] $k \geq 3$). This problem is reduced by two iterations of Lagrangian Relaxation to a [2-dimensional] two-dimensional penalized cost function $\boldsymbol{\phi}_j^{(k-1)_i}$. This cost function, and all other cost functions described below, are also non-smooth and continuous but are illustrated in Figure 6 as smooth for explanation purposes. Solving the $\boldsymbol{\phi}_j^{(k-1)_i}$ problem results in one set of z^2 assignments and the value of $\Phi_{(k-1)_i}$ at the point u_{ij}^2. A series of functions $\boldsymbol{\phi}_j^{(k-1)_i},\ldots,\boldsymbol{\phi}_1^{(k-1)_i}$ each generated from a different u^3 are shown. The particular series illustrates the process of locating the peak of $\Phi_{(k-1)_i}$. The [2-dimensional] two-dimensional penalized cost functions $\boldsymbol{\phi}_j^{(k-1)_i},\ldots,\boldsymbol{\phi}_1^{(k-1)_i}$ can be solved directly. Each such solution provides the information required to calculate the next u^3 value. Each iteration of the hill climbing improves the selection of u^3 values, i.e., yields ϕ^2 problem whose solution is closer to those at the solution of the ϕ^3 problem. The result of solving $\boldsymbol{\phi}_j^{(k-1)_i}$ is values that are on both $\Phi_{(k-1)_i}$ and Φ_k [Φ_k]. Figure 6 illustrates the surface Φ_k which comprises the minimums of all k-dimensional penalized cost function surfaces ϕ_k, i.e., if the Problem Formulations (1.5) and [[3.4]] (1.7) were solved at all possible values of u^k the

function Φ_k [$_k$ Φ_k] (u_k) would result. The [Φ_k]surface Φ_k is always less than the [v_k]surface v_k except at the peak as described in Equation [[3.5]] (1.8) and its maximum occurs where the [v_k]surface v_k is minimum. Because the [Φ_k Φ_k v_k the surface. The Φ_k] Φ_k represents the minimum[s] of the surface Φ_k, any point on the surface Φ_k can mapped to the surface v_k. The function Φ_k provides a lower bound on the minimization problem described by [[3.1] (a)] (1.4)(a). Let $\bar{\mathbf{z}}^k$ be the unknown solution to Problem Formulation [[3.1]] (1.4) and note that:

[[3.5]] (1.8)

$$\Phi^k(u^k) \le v_k(\bar{z}^k) \le v_{k-1}(z^k)$$

Consequently, the [z^k]values z^k at the peak of Φ_k (i.e., the greatest lower bound on the cost of the relaxed problem), can be substituted into the k-dimensional penalized cost function to determine the proper assignments. Consequently, the present invention attempts to find the maximum on the [Φ_k]surface Φ_k. However, it is not possible to use the prior art hill climbing to hill climb to the peak of Φ_k because the definition of Φ_k requires exact knowledge of lower order [Φ]functions Φ. As the solution of ϕ_1^k is not the exact[the] solution, in that higher order [u]values of u are not yet optimized, its value can be larger than the true peak of Φ_k. As such it is not a lower bound on v_k and it can not be used to recover the required solution.

Amend the paragraph beginning at column 22, line 7, as follows:

Instead, the present invention defines all auxiliary functions Ψ_k which [is] are based only on the solution to the [2-dimensional] two-dimensional penalized cost function problem, lower order values z^k and values [and]u^k [values] determined

previously by the reduction process. The [Ψ_k] function $\mathit{\Psi}_k$ is a less than approximation of $\mathit{\Phi}_k$, and its gradient is used for hill climbing to its peak. The [z^k] values z^k at the peak of the [Ψ_k] function $\mathit{\Psi}_k$ are then substituted into Problem Formulation [[3.2]] (1.5) to determine the proper assignments. To define the [Ψ_k]function $\mathit{\Psi}_k$, the present invention explicitly makes the function $\mathit{\Phi}_k(u^k)$ a function of all higher order sets of Lagrangian Coefficients with the expanded notation: [Φ_k U^k u^{k+1} u^k Ψ] $\mathit{\Phi}_k(u^k; u^{k+1},\ldots,u^k)$. Then, a new set of functions $\mathit{\Psi}_k$ is defined recursively, using the [Φ_k] $\mathit{\Phi}_k$'s domain:

[[3.6](a)] (1.9)

$$\Psi_3(u^3) = v_2 + \sum_{i_3=0}^{N_3} u^3_{i_3} = \Phi_3(u^3; u^4,\ldots,u^K)$$

where v_2 is the solution value for the most recent [2-dimensional] two-dimensional penalized cost function problem. For [k>3] $k > 3$

[[3.6](b)] (1.10)

$$\Psi_k(u^3,\ldots,u^{k-1};u^k) = \begin{cases} \Phi_k(u^k; u^{k+1},\ldots,u^K), \\ \qquad\qquad \textit{if known}, \\ \Psi_{k-1}(u^3,\ldots,u^{k-2};u^{k-1}) + \sum_{i_k=0}^{N_k} u^k_{i_k}, \\ \qquad\qquad \textit{otherwise}. \end{cases}$$

From the definitions of Φ_k [$_k$] and v_k (Problem Formulation (1.7) compared with Problem Formulation [[3.2]] (1.5)):

$$\Phi_k(u^k;u^{k+1},\dots,u^M) = v_{k-1}(z^{k-1}) + \sum_{i_k=0}^{N_k} u^k_{i_k}$$

it follows that:

$$\Psi_3(u^3) = v_2 + \sum_{i_3=0}^{N_3} u^3_{i_3} = \Phi_3(u^3;u^4,\dots,u^M) \leq v^3_3(z^3)$$

and with that Equation [3.5] (1.8) is extended to:

$$\Psi_k(u^3,\dots,u^{k-1};u^k) \leq \Phi_k(u^k;u^{k+1},\dots,u^M) \leq v_k(\bar{u}^k) \leq v_k(u^k)$$

[[3.7]] (1.11)

This relationship means that either Φ_k or Ψ_k may be used in hill climbing to update the Lagrangian Coefficients u. Φ_k is the preferred choice, however it is only available when the solution to Problem Formulation [[3.2]] (1.5) is a feasible solution to Problem Formulation [[3.1]] (1.4) (as in hill climbing from the second to third dimension which is why prior art worked). For simplicity in implementation, the function Ψ_k is defined so that it equals [Φ] Φ_k when either function could be used. It is therefore always used, even for hill climbing from the second dimension. The use of the [Ψ]function Ψ which is based on previously assigned or determined u values and not higher order u values which are not yet determined, is an important feature of the present invention.

Delete the paragraph beginning at column 23, line 1, thru column 23, line 14.

Amend the paragraph beginning at column 23, line 15, as follows:

I.2.2. Determining Effective Gradient

After the function Ψ_k is defined, the next steps of hill climbing/peak detection are to determine the gradient of the [Ψ_k]function Ψ_k, determine an increasing portion of the gradient and then move up the [Ψ_k]surface Ψ_k in the direction of this increasing portion of the gradient. As shown in Figure 6, any upward step on the [Ψ_k the Φ_{kj} U^k α α] surface Ψ_k, for example, to the minimum of the $\phi^k{}_j$ will yield a new set of values u^k (to the "left") that is closer to the ideal set of values u^k[values] which correspond to the minimum of the [v_k]function v_k. While it is possible to iteratively step up this [Ψ_k]surface Ψ_k with steps of fixed size and then determine if the peak has been reached, the present invention optimizes this process by determining the single step size from the starting point at the minimum of [ϕ_{ki}]$\phi^k{}_i$ that will jump to the peak and then [calculating]calculate the values u^k[values] at the peak. Once the [f."u" s]values u at the peak [at]of Ψ_k are determined, then the values u^k[values] can be substituted into Problem Formulation [[3.2]] (1.5) to determine the proper assignment. (However, in a more realistic example, where *k* is much greater than three, then the values u^k[values] at the peak of the [Ψ]function Ψ along with the [lower order] lower-order values u^k[values] and those assigned and yielded by the reduction steps are used to define the next higher level function Ψ. This definition of a higher order function Ψ and hill climbing process are repeated iteratively until Ψ_k, the peak of Ψ_k, and the values u^k at the peak of Ψ_k are identified.) The following is a description of how to determine the gradient of each [Ψ]surface Ψ and how to determine the single step size to jump to the peak from the starting point on each surface Ψ.

Delete the paragraph beginning at column 23, line 32, thru column 23, line 54.

Amend the paragraph beginning at column 23, line 55, as follows:

As noted above, each surface Ψ is non-smooth. Therefore, if a gradient was taken at a single point, the gradient may not point toward the peak. Therefore, several gradients (a "bundle") are determined at several points on the [Ψ_k]surface Ψ_k in the region of the starting point (i.e., minimum of ϕ_{k_i}) and then averaged. Statistically, the averaged result should point toward the peak of the [Ψ_k]surface Ψ_k. Wolfe's Conjugate Subgradient Algorithm ([Wolfe75, Wolfe79]P. Wolfe. A method of conjugate subgradients for minimizing non-differentiable functions. *Mathematical Programming Study*, 3:145-173, 1975; P. Wolfe. Finding the nearest point in a polytope. *Mathematical Programming Study*, 11:128-149, 1976) for minimization was previously known in another environment to determine a gradient of a non-smooth surface using multiple subgradients and can be used with modification in the present invention. [Wolfe's algorithm is further described in "A Method of Conjugate Subgradients for Minimizing Nondifferentiable Functions" page 147-173 published by Mathematical Programming Study 3 in 1975 and "Finding the Nearest Point in a Polytope" page 128-149 published by Mathematical Programming Study 11 in 1976.] The modification to Wolfe's algorithm uses the information generated for $\Psi_k(u^{k3},\ldots,u^{kk-2};\ u^{kk-1})$ as the basis for calculating the subgradients. The definition of a subgradient v of $\Psi_k(u^{k3},\ldots,u^{kk})$ is any member of the subdifferential set defined as:

$$[\delta\Psi_k(u)=\{v\in R^{N_k+1}|\,(\Psi_k(u^{k3},\ldots,u^{kk-1};u')-\Psi_k(u^{k3},\ldots,u^{k-1};u^k))\geq v^T(u'-u^k)\qquad\forall\ u'\in R^{N_k+1}\}]$$

$$\partial\Psi_k(u)=\{v\ \in\ R^{N_k+1}|\,(\Psi_k(u^{k3},\ldots,u^{kk-1};u')-\Psi_k(u^{k3},\ldots,u^{k-1};u^k))\geq v^T(u'-u^k)\quad\forall\ u'\in\ R^{N_k+1}\},$$

[(where v^T is the transpose of v)]where v^T is the transpose of v.

Next, a subgradient vector is determined from this function. If z^k is the solution of Problem Formulation [[3](1.5), then differentiating $\Psi_k(u^3, \ldots, u^{k-1}; u^k)$ with respect to u^k_{ik} and evaluating the result with respect to the current selection matrix z^k yields a subgradient vector:

$$g=(0,(1-\sum_{i_1=0}^{N_1} \ldots \sum_{i_{k-1}=0}^{N_k} z^k_{i_1,\ldots i_k} \mid i_k=1,\ldots,N_k))$$

Amend the paragraph beginning at column 24, line 19, as follows:

The iterative nature of the solution process at each dimension yields a set of such subgradients. Except for a situation described below, where the resultant averaged gradient does not point toward the peak, the most recent set of such subgradients are saved and used as the "bundle" for the peak finding process for this dimension. For example, at the [k] level k there is a bundle of subgradients of the [Ψ_k]surface Ψ_k near the minimum of the [ϕ_{ki}]surface Φ^k_i determined as a result of solving Problem Formulation [[3.1]](1.4) at all lower levels. This bundle can be averaged to approximate the gradient. Alternately, the previous bundle can be discarded so as to use the new value to initiate a new bundle. This choice provides a way to adjust the process to differing classes of problems, i.e., when data is being derived from two sensors and the relaxation proceeds from data derived from one sensor to the other then the prior relaxation data for the first sensor could be detrimental to performance on the second sensors data.

Amend the paragraph beginning at column 24, line 37, as follows:

I.2.3. Step Size and Termination Criterion

After the average gradient of the [Ψ_k]surface Ψ_k is determined, the next step is to determine a single step that will jump to the peak of the [Ψ_k]surface Ψ_k. The basic strategy is [to]first to specify an arbitrary step size, and then calculate the value of Ψ_k at this step size in the direction of the gradient. If the [Ψ_k]value of Ψ_k is larger than the previous one, this probably means that the step has not yet reached the peak. Consequently, the step size is doubled and a new [Ψ_k]value of Ψ_k is determined and compared to the previous one. This process is repeated until the new [Ψ_k]value of Ψ_k is less than the previous one. At that time, the step has gone too far and crossed over the peak. Consequently, the last doubling is rolled-back and that step size is used for the iteration. If the initial estimated Ψ_k is less than previous value then the step size is decreased by 50%. If this still results in a smaller [Ψ_k]value of Ψ_k, then the last step is rolled back and the previous step size is decreased by 25%. The following is a more detailed description of this process.

Amend the paragraph beginning at column 24, line 51, as follows:

With a suitable bundle of subgradients determined as just described[.], Wolfe's algorithm can be used to determine the effective subgradient [(d)] d and the [Upgraded]upgraded value u^k_{j+1}. From the previous iteration, or from an initial condition, there exists a step length value [*(t)*] t. The value,

$$\mathbf{u}_+ = \mathbf{u}^k_j + t\mathbf{d}$$

is calculated as an estimate of u^k_{j+1}. To determine if the current step size is valid [the] we evaluate $\Psi_k(u^3, \ldots, u^{k-2}; u_+)$. If the

result represents an improvement then we double the step size. Otherwise we halve the step size. In either case a new u_+ is calculated. The doubling or halving continues until the step becomes too large to improve the result, or until it becomes small enough to not degrade the result. The resulting suitable step size is saved with d as part of the subgradient bundle. The last acceptable u_+ is assigned to u^k_{j+1}.

Amend the paragraph beginning at column 25, line 1, as follows:

Three distinct [criterion]<u>criteria</u> are used to determine when u^k_j is close enough to $\overline{u}^k$:

1. The Wolfe's algorithm criterion of **d**=0 given that the test has been repeated with the bundle containing only the most recent subgradient.
2. The difference between the lower bound $\Phi_k(\mathbf{u}^k)$ and the upper bound $v_k(\mathbf{z}^k, \mathbf{u}^k)$ being less than a preset relative threshold. (Six percent was found to be an effective threshold for radar tracking problems.)
3. An iteration count being exceeded.

Amend the paragraph beginning at column 25, line 11, as follows:

The use of limits on the iteration are particularly effective for iterations at the level 3 through <u>(</u>n-1[levels,]<u>)</u> as these iterations will be repeated so as to resolve higher order coefficient sets. With limited iterations the process is in general robust enough to improve the estimate of upper order Lagrangian Coefficients. By limiting the iteration counts then the total processing time for the algorithm becomes deterministic. That characteristic means the process can be effective for real time problems such as radar, where the results of the last scan of the environment must be processed prior to

the next scan being received.

Amend the paragraph beginning at column 25, line 22, as follows:

I.2.4. Solution Recovery

The following process determines if the u^k values at what is believed to be the peak of the [Ψ_k]function Ψ_k adequately approximate the u^k values at the peak of the corresponding [Φ_k]function Φ_k. This is done by determining if the corresponding penalized cost function is minimized. Thus, the u^k values at the peak of the [Ψ_k]function Ψ_k are first substituted into Problem Formulation [[3.2]](1.5) to determine a set of z assignments for k-1 points. During the foregoing reduction process, Problem Formulation [[3.4]](1.7) yielded a tentative list of k selected z points that can be described by their indices as: [$\left\{\left(i_1^j \ldots i_{k-1}^j\right)\right\}_{j=1}^{N_0}$]$\underline{\left\{\left(i_1^j \ldots i_{k-1}^j\right)\right\}_{j=1}^{N_0}}$, where N_0 is the number of cost elements selected into the solution. One possible solution of Problem Formulation [[3.1]](1.4) is the solution of Problem Formulation [[3.2]](1.5) which is described as[$\left\{\left(i_1^j \ldots i_{k-1}^j m_{i_1 \ldots i_{k-1}}^k\right)\right\}_{j=1}^{N_0}$]$\underline{\left\{\left(i_1^j \ldots i_{k-1}^j m_{i_1 \ldots i_{k-1}}^k\right)\right\}_{j=1}^{N_0}}$ with $m_{i_1 \ldots i_{k-1}}^k$ as it was defined in Problem Formulation [[3.3]](1.6). If this solution satisfies the [k-th]$\underline{k^{th}}$ constraint set, then it is the optimal solution.

Amend the paragraph beginning at column 25, line 39, as follows:

However, if the solution for Problem Formulation [[3.2]](1.5) is not feasible (decision 355), then the following adjustment process determines if a solution exists which satisfies the [k-th]$\underline{k^{th}}$ constraint while retaining the assignments made in solving Problem Formulation [[3.4]](1.7). To do this, a [2-dimensional]<u>two-dimensional</u> cost matrix is defined based upon all observations from the [k-th]$\underline{k^{th}}$ set which could be used to extend the relaxed solution[.]:

[[3.8]]

$$h_{jl}=c^{k}_{i_1,\dots,i_{k-1},l} \qquad \begin{array}{l} \textit{for } l=0,\dots,N_k \\ \textit{and } j=0,\dots,N_0. \end{array} \qquad \underline{(1.12)}$$

If the resulting [2-dimensional]two-dimensional assignment problem,

[3.9]]

$$\text{Minimize[:]} \qquad \sum_{j=0}^{N_0}\sum_{l=0}^{N_l} h_{jl}w_{jl}$$

$$\text{Subject to:} \qquad \sum_{l=0}^{N_k} w_{jl}=1 \qquad j=1,\dots,N_0, \qquad \underline{(1.13)}$$

$$[]\sum_{l=0}^{N_k} w_{jl}=1 \qquad j=1,\dots,N_0,$$

$$w_{jl}\in\{0,1\} \qquad j=0,\dots,N_0,\ l=0,\dots,N_k$$

has a feasible solution, then the indices of that solution map to the solution of Problem Formulation [[3.1]]<u>(1.4)</u> for the k-dimensional problem. The first index in each resultant solution entry is the pointer back to an element of the $\left\{\left(i_1^j \dots i_{k-1}^j m_{i_1 \dots i_{k-1}}\right)\right\}_{j=1}^{N_0}$ list. That element supplies the first k-1 indices of the solution. The second index of the solution to the recovery problem is the k^{th} index of the solution. Together these indexes specify the values of z^k that solve Problem Formulation [[3.1]]<u>(1.4)</u> at the k^{th} level.

Amend the paragraph beginning at column 26, line 8, as follows:

If Problem Formulation [[3.9]]<u>(1.13)</u> does not have a feasible solution, then the value of u_i^k which was thought to represent u_p^k is not representative of the actual peak and further iteration at the k^{th} level is required. This decision represent

the branch path from step 214 and equivalent steps.

Amend the paragraph beginning at column 26, line 24, as follows:

In partitioning to groups no consideration is given to the actual cost values. The analysis depends strictly on the basis of two or more cost elements sharing the same specific axis of the cost matrix. In a [2-dimensional]<u>two-dimensional</u> case two cost elements must be in the same group if they share a row or a column. If the two elements are in the same row, then each other cost element that is also in the same row, as well as any cost elements that are in columns occupied by members of the row must be included in the group. The process continues recursively. In literature it is referred to as "Constructing a Spanning Forest." The *k*-dimensional case is analogous to the [2-dimensional]<u>two-dimensional</u> case. The specific method we have incorporated is a depth first search, presented by Aho, Hopecraft, and Ullman<u> (A.V. Aho</u>, [in "]<u>J.E. Hopcroft, and J.D. Ullman. </u>*Design and Analysis of Computer Algorithms[," section 5.2, published by Addison-Westley]*<u>. Addison-Wesley</u>, MA<u>, 1974)</u>.

Amend the paragraph beginning at column 26, line 47, as follows:

The result of partitioning at level M is the set of problems described as $\{P_{ij}|i=1,\ldots,p_j$ and $j=1,\ldots,N\}$, where N is the total number of hypothesis. The problems labeled [$\{P_{ij}|i=1,\ldots,p_j$ and $j=1,\ldots,N\}$] <u>$\{P_{i1}|i=1,\ldots,p_1\}$</u> are the cases where [their]<u>there</u> is only one choice for the next observation at each scan and that observation could be used for no other track, i.e., it is a single isolated track. The hypothesis must be included in the solution set and no further processing is required.

Amend the paragraph beginning at column 26, line 48, as follows:

As [hypothesis] hypotheses are constructed the first element is used to refer to the track [id]ID. Any problem in the partitioned set which does not have shared costs in the first scan represent a case where a track could be extended in several ways but none of the ways share an observation with any other track. The required solution hypothesis for this case is the particular hypothesis with the maximum likelihood. For this case all likelihoods are determined as was described in Scoring and the maximum is selected.

Amend the paragraph beginning at column 26, line 57, as follows:

In addition to partitioning at the [M]level *M*, partitioning is applied to each subsequent level *M*-1,...,2. For each problem that was not eliminated by [either by]the prior selection, partitioning is repeated, ignoring observations that are shared in the last set of observations. Partitioning recognizes that relaxation will eliminate the last constraint set and thus partitioning is feasible for the lower level problems that will result from relaxation. This process is repeated for all levels down to k=3. The full set of partitionings can be performed in the Problem Formulation Step 310, prior to initiating the actual relaxation steps. The actual [2-dimensional]two-dimensional solver used in step 206 includes an equivalent process so no advantage would be gained by partitioning at the k=2 level.

Amend the paragraph beginning at column 27, line 4, as follows:

There are two possible solution methods for the remaining problems. "Branch and Bound" as was previously described, or the

relaxation method that this invention describes. If any of the partitions have 5-10 possible tracks and less than 50 to 20 hypotheses, then the prior art "Branch and Bound" algorithm generally executes faster than does the relaxation due to its reduced level of startup overhead. The "Branch and Bound" algorithm is executed against all remaining *M* level problems that satisfy the size constraint. For the remainder the Relaxation algorithm is used. The scheduling done in Problem Formulation allows each Problem Formulation [[3.2]](1.5) cost matrix resulting from the first step of relaxation to be partitioned. The resulting partitions can be solved by any of the four methods[,]: isolated track direct inclusion, isolated track tree evaluation, small group "Branch and Bound" or an additional stage of relaxation as has been fully described.

Amend the paragraph beginning at column 27, line 29, as follows:

The following is a more detailed description of the partitioning method. Its application at all but the [M]level *M* depends upon the relaxation process described in this invention. The recursive partitioning is therefore a distinct part of this invention. The advantage of this method is greatly enhanced by the sparse cost matrix resulting from tracking problems. However the sparse nature of the problem requires special storage and search techniques.

Amend the paragraph beginning at column 27, line 37, as follows:

A hypothesis is the combination of the cost element c_n the selection variable z_n and all observations that made up the potential track extension. It can be written as,
[$h_n=\{c_n, z_n, \{o_{n_k}=o_{k_i} | k=1,...,Mi \in \{1,...,N_k\}\}\}$]
$h_n=\{c_n, z_n, \{o_{n_k}=o_{k_i} | k=1,...,M_i \in \{1,...,N_k\}\}\}$, i.e., cost, selection

variable and observations in the hypothesis. [n∈{1,...,*N*}]Here *n* ∈{1,...,*N*}, where *N* is the total number of hypotheses in the problem. While the cost and assignment matrices were previously referenced, these matrices are not effective storage mechanisms for tracking applications. Instead the list of all hypothesis and sets of lists for each dimension that reference the hypothesis set are stored. The hypothetical set in list form is:

$$\{h_n\}_{n=1}^{n=N}$$

For each dimension [k=1]*k*=1,...,*M* there exists a set of lists, with each list element being a pointer to a particular hypothesis:

$$L_{k_i} = \{p_{k_j} | k_j = 1,\ldots,N_{k_i}\}_{i=1,k=1}^{i=N_k,k=M}$$

where N_{k_i} is a number of hypothesis containing the [i-th] i^{th} observation from scan *k*. This structure is a multiply linked list in that any observation is associated with a set of pointer to all hypothesis it participates in, and any hypothesis has a set of pointers to all observations that formed it. (These pointers can be implemented as true pointers or indices depending upon the particular programming language utilized.)

Amend the paragraph beginning at column 27, line 62, as follows:

Given this description of the matrix storage technique then the partitioning technique is as follows: Mark the first list L_{k_i} and follow out all pointers in that list to the indicated hypothesis $h_{p_{k_i}}$ for i=1,..., [N_{ki}] N_{k_i} . Mark all located hypothesis, and for each follow pointers back the particular [L_{ok}]L_k for k=1,...,M. Those L's if not previously marked get

marked and also followed out to hypothesis elements and back to *L*'s. When all such *L*'s or *h*'s being located are marked, then an isolated sub-problem has been identified. All marked elements can be removed from the lists and stored as a unique problem to be solved. The partitioning problem then continues by again starting at the first residual [L]set *L*. When none remain, the original problem is completely partitioned.

Amend the paragraph beginning at column 28, line 11, as follows:

Isolated problems can result from one source track having multiple possible extensions or from a set of source tracks contending for some observations. Because one of the indices of *k* (in our implementation it is *k*=1) [indicate] indicates the source track then it is possible to categorize the two problem types by observing if the isolated problem includes more than one *L*-list from the index level associated with tracks.

Amend the paragraph beginning at column 28, line 62, as follows:

Based on the foregoing, apparatus and methods have been disclosed for tracking objects. However, numerous modifications and substitutions can be made without deviating from the scope of the present invention. For example, the foregoing [Ψ]functions Ψ can also be defined as recursive approximation problem in which several values of higher order u^k values are used to eliminate the higher than approximation characteristic of the [Φ_k

Ψ

Φ_k]

function Φ_k. The hill climbing of the function Ψ can be implemented by a high order hill climbing using the enhanced function Φ_k. Although the result would not be as efficient it seems likely that the method would converge. Therefore, the

invention has been disclosed by way of example and not limitation, and reference should be made to the following claims to determine the scope of the present invention.

Amend the paragraph beginning at column 29, line 25, as follows:

II.1. Formulation of the Assignment Problem

The goal of this section is to explain the formulation of the data association problems, and more particularly multi-dimensional assignment problems, that govern large classes of problems in centralized or hybrid centralized-sensor level multisensor/multitarget tracking. The presentation is brief; technical details are presented for both track initiation and maintenance in (A.B. Poore[,]. [Multi-dimensional] Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion[,]. *Computational Optimization and Applications*, 3[(1994), pp.]:27-57, 1994) for [nonmaneuvering] non-maneuvering targets and in [Ingemar J. Cox, Pierre Hansen, and Beta Julesz, eds., *Partitioning Data Sets*,](A.B. Poore. Multidimensional assignments and multitarget tracking: Partitioning data sets. In P. Hansen, I.J. Cox, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 19, pages 169-198, Providence, R.I., 1995. American Mathematical Society[, Providence, R.I., v. 19, Feb. 1995, pp. 169-198]) for maneuvering targets. Note that the present formulation can also be modified to include target features (e.g., size and type) into the scoring process 154.

Amend the paragraph beginning at column 29, line 43, as follows:

The data assignment problems for multisensor and multitarget tracking considered here are generally posed as that of maximizing the posterior probability of the surveillance region

(given the data) according to

[[5.1]]

$$\textit{Maximize} \left\{ \frac{P(\Gamma=\gamma \mid Z^M)}{P(\Gamma=\gamma^0 \mid Z^M)} \mid \gamma \in \Gamma^* \right\}, \qquad \underline{(2.1)}$$

where Z^M represents M data sets, γ is a partition of indices of the data (and thus induces a partition of the data), Γ^* is the finite collection of all such partitions, [1 $*\gamma^0$] Γ is a discrete random element defined on Γ^*, γ^0 is a reference partition, and [$P(\Gamma=\gamma|Z^M)$] $P(\Gamma=\gamma|Z^M)$ is the posterior probability of a partition γ being true given the data Z^M. The term partition is defined below.

Amend the paragraph beginning at column 29, line 59, as follows:

Consider M observation sets $Z(k)$ (k=1,...,N) each of N_k observations $\{z_{i_k}^k\}_{i_k=1}^{N_k}$, and let Z^M denote the cumulative data set defined by

$$Z(k) = \{z_{i_k}^k\}_{i_k=1}^{N_k} \text{ and } Z^M = \{Z(1),...,Z(M)\},$$

[5.2]

$$Z(k) = \left\{ z_{i_k}^k \right\}_{i_k=1}^{N_k} \textit{ and } Z^m = \left\{ Z(1),...,Z(M) \right\}, \qquad \underline{(2.2)}$$

respectively. In multisensor data fusion and multitarget tracking the data sets Z(k) may represent different classes of objects, and each data set can arise from different sensors. For track initiation the objects are measurements that must be partitioned into tracks and false alarms. In the formulation of track extensions a moving window over time of observations sets is used. The observation sets will be measurements which are: (a) assigned to existing tracks, (b) designated as false

measurements, or (c) used for initiating tracks. However, note that alternative data objects instead of observation sets may also be fused such as in sensor level tracking wherein each sensor forms tracks from its own measurements and then the tracks from the sensors are fused in a central location. Note that, as one skilled in the art will appreciate, both embodiments of the present invention may be used for this type of data fusion.

Amend the paragraph beginning at column 30, line 14, as follows:

The data assignment problem considered presently is represented as a case of set partitioning defined in the following way. First, for notational convenience in representing tracks, a zero index is added to each of the index sets in [[5.2]]<u>(2.2)</u>, a gap filler z_0^k is added to each of the data sets $Z(k)$ in [[5.2]]<u>(2.2)</u>, and a "track of data" is defined as $\left(z_{i_1}^1,\ldots,z_{i_N}^N\right)$ <u>,</u> where i_k [and z_k] can now assume the values of 0[and z, respectively]. A partition of the data will refer to a collection of tracks of data or track extensions wherein each observation occurs exactly once in one of the tracks of data and such that all data is used up. Note that the occurrence of the gap filler is unrestricted. The gap filler z_0^k serves several purposes in the representation of missing data, false observations, initiating tracks, and terminating tracks. Note that a "reference partition" may be defined which is a partition in which all observations are declared to be false.

Amend the paragraph beginning at column 30, line 31, as follows:

Next under appropriate independence assumptions it can be shown that:

[[5.3]]

$$\frac{P(\Gamma=\gamma|Z^M)}{P(\Gamma=\gamma^0|Z^M)} \equiv L_\gamma \equiv \prod_{(i_1 \ldots i_N)\in\gamma} L_{i_1 \ldots i_M}\,, \qquad \underline{(2.3)}$$

where $L^k_{i_1 \ldots i_M}$ is a likelihood ratio containing probabilities for detection, maneuvers, and termination as well as probability density functions for measurement errors, track initiation and termination. Then if [$c^k_{i_1 \ldots i_N}=-\ln(L^k_{i_1 \ldots i_M})$]
$\underline{c^k_{i_1 \ldots [iN=-ln] i_N} = -\ln(L^k_{i_1 \ldots i_M}),}$

[[5.4]]

$$-\ln\left[\frac{P(\Gamma=\gamma|Z^M)}{P(\Gamma=\gamma^0|Z^M)}\right] = \sum_{(i_1,\ldots,i_M)\in\gamma} c_{i_1 \ldots i_N}. \qquad \underline{(2.4)}$$

Delete the paragraph beginning at column 30, line 48, thru column 31, line 18, and replace it with the following new paragraph:

<u>Using (2.3) and the zero-one variable $z^k_{i_1 \ldots i_N}=1$ if $(i_1,\ldots,i_M)\in\gamma$ and 0 otherwise, one can then write the problem (2.1) as the following M-dimensional assignment problem (as also presented in section Problem Formulation (1.4) with $k=M$):</u>

(a) Minimize $$\sum_{i_1=0}^{N_1} \cdots \sum_{i_M=0}^{N_M} c_{i_1 \ldots i_M} z_{i_1 \ldots i_M}$$

(b) Subject to: $$\sum_{i_2=0}^{N_2} \cdots \sum_{i_M=0}^{N_M} z_{i_1 \ldots i_M} = 1, \qquad i_1 = 1, \ldots, N_1,$$

(c) $$\sum_{i_1=0}^{N_1} \cdots \sum_{i_{k-1}=0}^{N_{k-1}} \sum_{i_{k+1}=0}^{N_{k+1}} \cdots \sum_{i_M=0}^{N_M} z_{i_1 \ldots i_M} = 1, \qquad (2.5)$$

for $i_k = 1, \ldots, N_k$ and $k=2, \ldots, M-1$,

(d) $$\sum_{i_1=0}^{N_1} \cdots \sum_{i_{M-1}=0}^{N_{M-1}} z_{i_1 \ldots i_M} = 1, \qquad i_M = 1, \ldots, N_M$$

(e) $$z_{i_1 \ldots i_M} \in \{0,1\} \qquad \text{for all } i_1, \ldots, i_M$$

where $c_{0 \ldots 0}$ is arbitrarily defined to be zero. Here, each group of sums in the constraints (2.5)(b)-(2.5)(e) represents the fact that each non-gap filler observation occurs exactly once in a "track of data." One can modify this formulation to include multiassignments of one, some, or all the actual observations. The assignment problem (2.5) is changed accordingly. For example, if $z_{i_k}^k$ is to be assigned no more than, exactly, or no less than $n_{i_k}^k$ times, then the "=1" in the appropriate one of the constraints [5.5](b)-[5.5](d) is changed to "$\leq, =, \geq n_{i_k}^k$," respectively.

Amend the paragraph beginning at column 31, line 19, as follows:

Expressions for the likelihood ratios $L_{i_1 \ldots i_M}^k$[*such as [5.5]*] appearing in the costs $c_{i_1 \ldots i_M} = -\ln(L_{i_1 \ldots i_M})$ are [well known well-known. In particular, discussions of these ratios may be found in[:] (A.B. Poore[,]. [Multi dimensional] Multi-dimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion, *Computational Optimization*

and Applications[, 3 (1994), pp. 27-57; and Ingemar J. Cox, Pierre Hansen, and Beta Julesz, eds., Partitioning Data Sets,], 3:27-57, 1994; A.B. Poore. Multidimensional assignments and multitarget tracking: Partitioning data sets. In P. Hansen, I.J. Cox, B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 19, pages 169-198, Providence, R.I., 1995. American Mathematical Society[, Providence, R]). [I., v. 19, 1995, pp. 169-198.] Furthermore, the likelihood ratios are easily modified to include target features and to account for different sensor types. Also note that in track initiation, the M observation sets provide observations from M sensors, possibly all the same. Additionally note that for track maintenance, a sliding window of M observation sets and one data set containing established tracks may be used. In this latter case, the formulation is the same as above except that the dimension of the assignment problem is now $M+1$.

Amend the paragraph beginning at column 31, line 41, as follows:

Having formulated an M-dimensional assignment problem [[5.5]](2.5), we now turn to a description of the Lagrangian relaxation algorithm. Subsection II.2.1 below presents many of the relaxation properties associated with the relaxation of an n-dimensional assignment problem to an m-dimensional one via a Lagrangian relaxation of $n-m$ sets of constraints wherein [m<]$m \leq n \leq M$ and preferably in the present invention embodiment [n-m>1]$n-m \geq 1$. Although any $n-m$ constraint sets can be relaxed, the description here is based on relaxing the last $n-m$ sets of constraints and keeping the first m sets. Given either an optimal or suboptimal solution of the relaxed problem, a technique for recovering a feasible solution of the n-dimensional problem is presented in subsection II.2.2 below and

an overview of the Lagrangian relaxation algorithm is given in subsection II.2.3.

Amend the paragraph beginning at column 31, line 56, as follows:

The following notation will be used throughout the remainder of the work. Let M be an integer such that [M≥3]$\underline{M \geq 3}$ and let $n \in \{3,...,M\}$. The n-dimensional assignment problem is

[[6.1]]

$$\text{(a)}\quad \text{Minimize: } v_n(z) = \sum_{i_1=0}^{N_1} \cdots \sum_{i_n=0}^{N_n} c^n_{i_1 \ldots i_n} z^n_{i_1 \ldots i_n}$$

$$\text{(b)}\quad \textit{Subject to[:]} \quad \sum_{i_2=0}^{N_2} \cdots \sum_{i_n=0}^{N_n} z^n_{i_1 \ldots i_n} = 1, \quad i_1 = 1,\ldots,N_1, \qquad \underline{(2.6)}$$

$$\text{(c)}\quad \sum_{i_1=0}^{N_1} \cdots \sum_{i_{k-1}=0}^{N_{k-1}} \sum_{i_{k+1}=0}^{N_{k+1}} \cdots \sum_{i_n=0}^{N_n} z^n_{i_1 \ldots i_n} = 1,$$

for $i_k = 1,\ldots,N_k$ and $k=2,\ldots,n-1$,

$$\text{(d)}\quad \sum_{i_1=0}^{N_1} \cdots \sum_{i_{n-1}=0}^{N_{n-1}} z^n_{i_1 \ldots i_n} = 1, \qquad i_n = 1,\ldots,N_n,$$

$$\text{(e)}\quad z^n_{i_1 \ldots i_n} \in \{0,1\} \qquad \text{for all } i_1,\ldots,i_n$$

To ensure that a feasible solution of [[6.1]]$\underline{(2.6)}$ always exists, all variables with exactly one nonzero index (i.e., variables of the form $z^n_{0\ldots 0 i_k 0 \ldots 0}$ for $i_k \neq 0$) are assumed free to be assigned and the corresponding cost coefficients are well-defined.

Amend the paragraph beginning at column 32, line 23, as follows:

The *n*-dimensional assignment problem [[6.1]](2.6) has *n* sets of constraints. A (N_k+1)-dimensional multiplier vector associated with the [k-th]k^{th} constraint set will be denoted by $u^k=(u_0^k, u_1^k, \ldots, u_{M_k}^k)^T$ with $u_0^k=0$ and $k=1,\ldots,n$. The *n*-dimensional assignment problem [[6.1]](2.6) is relaxed to an *m*-dimensional assignment problem by incorporating $n-m$ of the *n* sets of constraints into the objective function [[6.1(a)]](2.6)(a). Although any constraint set can be relaxed, the description of the relaxation procedure for [[6.1]](2.6) will be based on the relaxation of the last $n-m$ sets of constraints. The relaxed problem is

$$\Phi_{mn}(u^{m+1},\ldots,u^n) \equiv \text{Minimize} \quad \phi_{mn}(z^n;u^{m+1},\ldots,u^n) \qquad [[6.2]](2.7)$$

$$\equiv \sum_{i_1=0}^{N_1}\cdots\sum_{i_n=0}^{N_n} c^n_{i_1\ldots i_n} z^n_{i_1\ldots i_n}$$

$$+ \sum_{k=m+1}^{n}\sum_{i_k=0}^{N_k} u^k_{i_k}\left[\sum_{i_1=0}^{N_1}\cdots\sum_{i_{k-1}=0}^{N_{k-1}}\sum_{i_{k+1}=0}^{N_{k+1}}\cdots\sum_{i_n=0}^{N_n} z^n_{i_1\ldots i_n}-1\right]$$

$$\equiv \sum_{i_1=0}^{N_1}\cdots\sum_{i_n=0}^{N_n}\left[c^n_{i_1\ldots i_n}+\sum_{k=m+1}^{n} u^k_{i_k}\right] z^n_{i_1\ldots i_n} - \sum_{k=m+1}^{n}\sum_{i_k=0}^{N_k} u^k_{i_k}$$

Subject to: $$\sum_{i_2=0}^{N_2}\cdots\sum_{i_n=0}^{N_n} z^n_{i_1\ldots i_n}=1, \qquad i_1 = 1,\ldots,N_1,$$

$$\sum_{i_1=0}^{N_1}\cdots\sum_{i_{k-1}=0}^{N_{k-1}}\sum_{i_{k+1}=0}^{N_{k+1}}\cdots\sum_{i_n=0}^{N_n} z^n_{i_1\ldots i_n}=1,$$

for $i_k = 1,\ldots,N_k$ and $k=2,\ldots,m$,

$$z^n_{i_1\ldots i_n} \in \{0,1\} \qquad \text{for all } i_1,\ldots,i_n.$$

wherein we have added the multiplier $u_0^k \equiv 0$ for notational

convenience. Thus, the multiplier $\mathbf{u}^k \in \mathbb{R}^{N_k+1}$ with $u_0^k=0$ is fixed.

Amend the paragraph beginning at column 32, line 62, as follows:

An optimal (or suboptimal) solution of [[6.2]]<u>(2.7)</u> can be constructed from that of an m-dimensional assignment problem. To show this, define for each $(i_1,...,i_m)$ an index $(j_{m+1},...,j_n) = (j_{m+1}(i_1,...,i_m),...,j_n(i_1,...,i_m))$ and a new cost function $c^m_{i_1...i_m}$ by

[[6.3]]<u>(2.8)</u>

[If $(j_{m+1},...,j_n)$]<u>If $(j_{m+1},...,j_n)$</u> is not unique, choose the smallest such j_{m+1}, amongst those $(n-m)$-tuples with the same j_{m+1} choose the smallest j_{m+2}, etc., so that $(j_{m+1},...,j_n)$ is uniquely defined.) Using the cost coefficients defined in this way, the following m-dimensional assignment problem is [obtaIned] <u>obtained</u>:

$\underline{\Phi}_{mn}(u^{m+1},...,u^n)$ = Minimize $\hat{\Phi}_{mn}(z^m;u^{m+1},...,u^n) \equiv v_m(z^m)$

Please replace the paragraph beginning on page 85, line 7, with the following amended paragraph:

[[6.4]]<u>(2.9)</u>

$$\Phi_{mn}(u^{m+1},...,u^n) = \text{Minimize } \hat{\Phi}_{mn}(z^m;u^{m+1},...,u^n) \equiv v_m(z^m)$$

$$\equiv \sum_{i_1=0}^{N_1} \cdots \sum_{i_m=0}^{N_m} c^m_{i_1 \ldots i_m} z^m_{i_1 \ldots i_m}$$

Subject to:

$$\sum_{i_2=0}^{N_2} \cdots \sum_{i_m=0}^{N_m} z^m_{i_1 \ldots i_m} = 1, \qquad i_1 = 1,...,N_1,$$

$$\sum_{i_1=0}^{N_1} \cdots \sum_{i_{k-1}=0}^{N_{k-1}} \sum_{i_{k+1}=0}^{N_{k+1}} \cdots \sum_{i_m=0}^{N_m} z^m_{i_1 \ldots i_m} = 1,$$

for $i_k = 1,...,N_k$ and $k=2,...,m-1$,

$$\sum_{i_1=0}^{N_1} \cdots \sum_{i_{m-1}=0}^{N_{m-1}} z^m_{i_1 \ldots i_m} = 1, \qquad i_m=1,...,N_m$$

$$z^{m}_{i_1 \ldots i_m} \in \{0,1\} \qquad \text{for all } i_1, \ldots, i_m.$$

As an aside, observe that any feasible solution z^n of [[6.1]](2.6) yields a feasible solution z^m of [[6.4]](2.9) via the construction

$$z^{m}_{i_1 \ldots i_m} = \begin{cases} 1 & \text{if } z^{n}_{i_1 \ldots i_{n-1} i_n} = 1 \text{ for some } (i_{m+1}, \ldots, i_n) \\ 0 & \text{otherwise.} \end{cases}$$

Thus, the m-dimensional assignment problem [[6.4]](2.9) has at least as many feasible solutions of the constraints as the original problem [[6.1]](2.6).

Amend the paragraph beginning at column 33, line 51, as follows:

The following Fact A.2 has been shown to be true. It states that an optimal solution of Problem Formulation [[6.2]](2.7) can be computed from that of Problem Formulation [[6.4]](2.9). Furthermore, the converse to this fact is provided in Fact A.3. Moreover, if the solution of either of these two problems [[6.4]](2.9) or [[6.4]](2.9) is ϵ-optimal (i.e., the objective function associated with the suboptimal solution is within "ϵ" of the objective function associated with the optimal solution), then so is the other.

Amend the paragraph beginning at column 33, line 60, as follows:

Fact A.2. Let w^m be a feasible solution to problem [[6.4]](2.9) and define w^n by

[6.5]]

$$w^{n}_{i_1 \ldots i_n} = w^{m}_{i_1 \ldots i_m} \qquad if \ (i_{m+1}, \ldots i_n) = (j_{m+1}, \ldots j_n) \ and$$
$$(i_1, \ldots, i_m) \neq (0, \ldots, 0)$$

$$w^n_{i_1\ldots i_n}=0 \qquad if\ (i_{m+1},\ldots i_n)\neq(j_{m+1},\ldots j_n)\ and \qquad \underline{(2.10)}$$
$$(i_1,\ldots,i_m)\neq(0,\ldots,0)$$

$$w^n_{0\ldots 0i_{m+1}\ldots i_n}=1 \qquad \text{if } c^n_{0\ldots 0i_{m+1}\ldots i_n} + \sum_{k=m+1}^{n} u^k_{i_k} \leq 0$$

$$w^n_{0\ldots 0i_{m+1}\ldots i_n}=0 \qquad if\ c^n_{0\ldots 0i_{m+1}\ldots i_n} + \sum_{k=m+1}^{n} u^k_{i_k} > 0\underline{.}$$

Then w^n is a feasible solution of the Lagrangian relaxed problem [[6.2]]<u>(2.7)</u> and

$$\Phi_{mn}(w^n;u^{m+1},\ldots,u^n) = \hat{\Phi}_{mn}(w^m;u^{m+1},\ldots,u^n)-\sum_{k=m+1}^{n}\sum_{i_k=0}^{M_k} u^k_{i_k}\ .$$

If, in addition, w^m is optimal for [[6.4]]<u>(2.9)</u>, then w^n is an optimal solution of [[6.2]]<u>(2.7)</u> and

$$\Phi_{mn}(u^{m+1},\ldots,u^n) = \hat{\Phi}_{mn}(u^{m+1},\ldots,u^n)-\sum_{k=m+1}^{n}\sum_{i_k=0}^{M_k} u^k_{i_k}\ .$$

Amend the paragraph beginning at column 34, line 25, as follows:

Fact A.3. Let w^n be a feasible solution to problem [[6.2]]<u>(2.7)</u> and define w^m by

[[6.6]]<u>(2.11)</u>

$$w^m_{i_1\ldots i_m}=\sum_{i_{m+1}=0}^{N_{m+1}}\cdots\sum_{i_n=0}^{N_m} w^n_{i_1\ldots i_n}\ for\ (i_1,\ldots,i_m)\neq(0,\ldots,0)\underline{,}$$

$$w^m_{0\ldots 0}=0 \text{ if } (i_1,\ldots,i_m)=(0,\ldots,0) \text{ and}$$

$$c^n_{0\ldots 0i_{m+1}\ldots i_n} + \sum_{k=m+1}^{n} u^k_{i_k} > 0 \text{ for all } (i_{m+1},\ldots,i_n)$$

$w^m_{0\ldots0}=1$ if $(i_1,\ldots,i_m)=(0,\ldots,0)$ and

$$c^n_{0\ldots0i_{m+1}\ldots i_n} + \sum_{k=m+1}^{n} u^k_{i_k} \le 0 \text{ for some } (i_{m+1},\ldots,i_n).$$

Then w^m is a feasible solution of the problem [[6.4]](2.9) and

$$\Phi_{mn}(w^n;u^{m+1},\ldots,u^n) \ge \hat{\Phi}_{mn}(w^m;u^{m+1},\ldots,u^n) - \sum_{k=m+1}^{n}\sum_{i_k=0}^{M_k} u^k_{i_k}$$

If, in addition, w^n is optimal for [[6.2]](2.7), then w^m is an optimal solution of [[6.4]](2.9),

$$\Phi_{mn}(w^n;u^{m+1},\ldots,u^n) = \hat{\Phi}_{mn}(w^m;u^{m+1},\ldots,u^n) - \sum_{k=m+1}^{n}\sum_{i_k=0}^{M_k} u^k_{i_k}, \text{ and}$$

$$\Phi_{mn}(u^{m+1},\ldots,u^n) = \hat{\Phi}_{mn}(u^{m+1},\ldots,u^n) - \sum_{k=m+1}^{n}\sum_{i_k=0}^{M_k} u^k_{i_k}.$$

Amend the paragraph beginning at column 34, line 52, as follows:

The next objective is to explain a recovery procedure or method for recovering a solution to the *n*-dimensional problem of [[6.1]](2.6) from a relaxed problem having potentially substantially fewer dimensions than [[6.1]](2.6). Note that this aspect of the alternative embodiment of the present invention is substantially different from the method disclosed in the first embodiment of the [multi dimensional] multi-dimensional assignment solving process of section I.1 of this specification. Further, this alternative embodiment provides substantial benefits in terms of computational efficiency and accuracy over the first embodiment, as will be discussed. Thus, given a feasible (optimal or suboptimal) solution w^m of [[6.4]](2.9) (or w^n if Problem Formulation [[6.2]](2.7) is constructed via Fact A.2), an explanation is provided here regarding how to generate a

feasible solution z^n of [[6.1]](2.6) which is close to w^m in a sense to be specified. We first assume that no variables in [[6.1]](2.6) are preassigned to zero (this assumption will be removed shortly). The difficulty with the solution w^n is that it need not satisfy the last $n-m$ sets of constraints in [[6.1]](2.6). Note, however, that if w^m is an optimal solution for [[6.4]](2.9) and w^n (constructed as in Fact A.2) satisfies the relaxed constraints, then w^n is optimal for [[6.1]](2.6). The recovery procedure described here is designed to preserve the 0-1 character of the solution w^m of [[6.4]](2.9) as far as possible. That is, if $w^m_{i_1 \ldots i_m}=1$ and $i_l \neq 0$ for at least one $l=1,\ldots,m$, then the corresponding feasible solution z^n of [[](2.6[.1]]) is constructed so that $z^n_{i_1 \ldots i_n}=1$ for some $(i_{m+1},\ldots,i_n)$. Note that by this reasoning, variables of the form $z^n_{0\ldots 0 i_{m+1}\ldots i_n}$ can be assigned to a value of 1 in the recovery problem only if $w^m_{0\ldots 0}=1$. However, variables $z^n_{0\ldots 0 i_{m+1}\ldots i_n}$ will be treated differently in the recovery procedure in that they can be assigned 0 or 1 independent of the value $w^m_{0\ldots 0}$. This increases the feasible set of the recovery problem, leading to a potentially better solution.

Amend the paragraph beginning at column 35, line 18, as follows:

Let $\{(i_1^j,\ldots,i_m^j)\}_{j=1}^{N_0}$ be an enumeration of indices of w^m (or the first m indices of w^n constructed as in Fact A.2) such that $w^m_{i_1^j \ldots i_m^j}=1$ and $(i_1^j,\ldots,i_m^j) \neq (0,\ldots,0)$. Set $(i_1^0,\ldots,i_m^0)=(0,\ldots,0)$ for j=0 and define

$$c^{n-m+1}_{j i_{m+1}\ldots i_n} = c^n_{i_1^j \ldots i_m^j i_{m+1}\ldots i_n} \qquad [[6.7]](2.12)$$

[for $i_k=0,\ldots,N_k$;
$k=m+1,\ldots,n$; $j=0,\ldots,N_0$.]

<u>for $i_k=0,\ldots,N_k$;</u>
<u>$k=m+1,\ldots,n$; $j=0,\ldots,N_0$.</u>

Let Y denote the solution of the (n-m+1)-dimensional assignment problem:

[[6.8]]<u>(2.13)</u>

$$\text{Minimize} \quad \sum_{j=0}^{N_0}\sum_{i_{m+1}=0}^{N_{m+1}}\cdots\sum_{i_n=0}^{N_m} c^{n-m+1}_{ji_{m+1}\cdots i_n} y_{ji_{m+1}\cdots i_n}$$

$$\textit{Subject to} \quad \sum_{i_{m+1}=0}^{N_{m+1}}\cdots\sum_{i_n=0}^{N_m} y_{ji_{m+1}\cdots i_n}=1, \qquad j=1,\ldots,N_0,$$

$$\sum_{j=0}^{N_0}\sum_{i_{m+1}=0}^{N_{m+1}}\cdots\sum_{i_{k-1}=0}^{N_{k-1}}\sum_{i_{k+1}=0}^{N_{k+1}}\cdots\sum_{i_n=0}^{N_m} y_{ji_{m+1}\cdots i_n}=1,$$

for $i_k=1,\ldots,N_k$ and $k=m+1,\ldots,n-1$,

$$\sum_{j=0}^{N_0}\sum_{i_{m+1}=0}^{N_{m+1}}\cdots\sum_{i_{n-1}=0}^{N_{m-1}} y_{ji_{m+1}\cdots i_n}=1, \qquad i_n=1,\ldots,N_m,$$

$$y_{ji_{m+1}\cdots i_n}\in\{0,1\} \quad \text{for all } j,\ i_{m+1},\ldots,i_n.$$

The recovered feasible solution z^n of [[6.1]]<u>(2.6)</u> corresponding to the multiplier set $\{u^m,\ldots,u^n\}$ is then defined by [[6.9]]

$$z^n_{i_1\cdots i_n} = \begin{cases} 1, & \textit{if } (i_1,\ldots,i_m)=(i_1^j,\ldots,i_m^j) \textit{ for some} \\ & j=0,\ldots,N_0 \textit{ and } Y_{ji_{m+1}}\ldots i_n=1, \\ 0, & \textit{otherwise.} \end{cases} \qquad \underline{(2.14)}$$

This recovery procedure is valid as long as all cost coefficients c^n are defined and all zero-one variables in z^n are free to be assigned. Note that modifications are necessary for sparse problems. If the cost coefficient

$[c^n_{i_1^j\ldots i_m^j i_{m+1}\ldots i_n}]$ <u>$c^n_{i_1^j\ldots i_m^j i_{m+1}\ldots i_n}$</u> is well defined and the zero-one variable

$[z^n_{i_1^j\ldots i_m^j i_{m+1}\ldots i_n}]\underline{c^n_{i_1^j\ldots i_m^j i_{m+1}\ldots i_n}}$ is free to be assigned to zero or one, then define $[c^{n-m+1}_{ji_{m+1}\ldots i_n} = c^n_{i_1^j\ldots i_m^j i_{m+1}\ldots i_n}]\underline{c^{n-m+1}_{ji_{m+1}\ldots i_n} = c^n_{i_1^j\ldots i_m^j i_{m+1}\ldots i_n}}$ as in [[6.7]]<u>(2.12)</u> with $z^{n-m+1}_{ji_{m+1}\ldots i_n}$ being free to be assigned. Otherwise, $z^{n-m+1}_{ji_{m+1}\ldots i_n}$ is preassigned to zero. To ensure that a feasible solution exists, we now only need ensure that the variables $z^{n-m+1}_{j0\ldots 0}$ are free for $j=0,1,\ldots,N_0$. (Recall that all variables of the form $z^n_{0\ldots i_k\ldots 0}$ are free (to be assigned) and all coefficients of the form $c^n_{0\ldots i_k\ldots 0}$ are well defined for $k=1,\ldots,n$.) This is accomplished as follows. If the cost coefficient $[c^n_{i_1^j i_2^j\ldots i_m^j 0\ldots 0}]\underline{c^n_{i_1^j i_2^j\ldots i_m^j 0\ldots 0}}$ is well defined and $[z^n_{i_1^j i_2^j\ldots i_m^j 0\ldots 0}]\underline{z^n_{i_1^j i_2^j\ldots i_m^j 0\ldots 0}}$ is free, then define $[c^{n-m+1}_{j0\ldots 0} = c^n_{i_1^j i_2^j\ldots i_m^j 0\ldots 0}]\underline{c^{n-m+1}_{j0\ldots 0} = c^n_{i_1^j i_2^j\ldots i_m^j 0\ldots 0}}$ with $z^{n-m+1}_{j0\ldots 0}$ being free. Otherwise, since all variables of the form $z^n_{0\ldots i_k\ldots 0}$ are known feasible and have well-defined costs, put

$$[c^{n-m+1}_{j0\ldots 0} = \sum_{k=1,i_k^j\neq 0}^{m} c^n_{0\ldots 0 i_k^j 0\ldots 0}]\underline{c^{n-m+1}_{j0\ldots 0} = \sum_{k=1,i_k^j\neq o}^{m} c^n_{0\ldots 0 i_k^j 0\ldots 0}}.$$

Amend the paragraph beginning at column 36, line 18, as follows:

Starting with the *M*-dimensional assignment problem [[6.1]]<u>(2.6)</u>, i.e., *n=M*, the algorithm described below is recursive in that the *M*-dimensional assignment problem is relaxed to an m-dimensional problem by incorporating (*n-m*) sets of constrains into the objective function using Lagrangian relaxation of this set. This problem is maximized with respect to the Lagrange multipliers, and a good suboptimal solution to the original problem is recovered using an (*n-m*+1)-dimensional assignment problem. Each of these two (the *m*-dimensional and the (*n-m*+1)-dimensional assignment problems) can be solved in a

similar manner.

Amend the paragraph beginning at column 36, line 29, as follows:

More precisely, reference is made to Figure 8 which presents a flowchart of a procedure embodying the multi-dimensional relaxation algorithm, referred to immediately above. This procedure, denoted MULTI_DIM_RELAX in Figure 8, has three formal parameters, *n*, PROB_FORMULATION and ASSIGNMT_SOLUTION, which are used to transfer information between recursive instantiations of the procedure. In particular, the parameter, *n*, is an input parameter supplying the dimension for the multi-dimensional assignment problem (as in [[6.1]](2.6)) which is to be solved (i.e., to obtain an optimal or near-optimal solution). The parameter, PROB_FORMULATION, is also an input parameter supplying the data structure(s) used to represent the *n*-dimensional assignment problem to be solved. Note that PROB_FORMULATION at least provides access to the cost matrix, [c^n], and the observation sets whose observations are to be assigned. Additionally, the parameter, ASSIGNMT_SOLUTION, is used as an output parameter for returning a solution of a lower dimensioned assignment problem to an instantiation of MULTI_DIM_RELAX which is processing a higher dimensioned assignment problem.

Amend the paragraph beginning at column 36, line 51, as follows:

A description of Figure 8 follows. Assuming MULTI_DIM_RELAX is initially invoked with M as the value for the parameter *n* and the PROB_FORMULATION having a data structure(s) representing an M-dimensional assignment problem as in [[5.51](2.5), in step 500 an integer *m*, [$2 \le m < n$]$2 \le m < n$ is chosen such that the constraint sets [or] corresponding to dimensions $m+1,\ldots,n$ are to be relaxed so that an m-dimensional problem formulation as in [[6.21](2.7)

results. In step 504 an initial approximation is provided for $\{u_0^{m+1},\ldots,u_0^n\}$. Subsequently, in step 508 the above initial values for $\{u_0^{m+1},\ldots,u_0^n\}$ are used in an iterative process in determining $(\underline{\bar{u}}^{l\ )m+1},\ldots,\underline{\bar{u}}^{l\ )n})$ which maximizes $\{\Phi_{mn}(u^{m+1},\ldots,u^n)$ where [$u^k \in \mathbb{R}^{Mk+1}$]$\underline{u^k \in \mathbb{R}^{M_{k+1}}}$ and $k=m+1,\ldots,n\}$ for a feasible solution w^n subject to the constraints of Problem Formulation [[6.2]]<u>(2.7)</u>. Note that by maximizing $\Phi_{mn}(u^{m+1},\ldots,u^n)$, some of the constraints that were relaxed are being forced to be satisfied and in so doing information is built into a solution of this function for solving the input assignment problem in "PROB_FORMULATION." Further note that a non-smooth optimization technique is used here and that a preferred method of determining a maximum in step 508 is the bundle-trust method of Schramm and Zowe [as described in]<u>(</u>H. Schramm and J. Zowe[,]<u>. </u>A version of the bundle idea for minimizing a non-smooth function: Conceptual idea, convergence analysis, numerical results[,]<u>. </u>*SIAM Journal on Optimization*, 2 [(1992)], [pp]<u>No</u>. <u>1:</u>121-152<u>, February, 1992)</u>. This method, along with various other methods for determining the maximum in step 508, are discussed below.

Amend the paragraph beginning at column 37, line 10, as follows:

Also note that for [m>2]$\underline{m > 2}$, a solution to the optimization problem of step 508 is NP-hard and therefore cannot be solved optimally. That is, there is no known computationally tractable method for guaranteeing an optimal solution. Thus, there are two possibilities: either (a) allow m to be greater than 2 and use auxiliary functions similar to those disclosed in the first embodiment of the k-dimensional assignment solver 300 in section I to compute a near-optimal solution, or (b) make $m=2$ wherein algorithms such as the forward/reverse auction algorithm of D.P. Bertsekas and D.A. Castañon [in their paper,]<u>(D.P. Bertsekas and D.A. Castañon. </u>A forward/reverse auction algorithm

for asymmetric assignment problems[,]. *Computational Optimization and Applications,* 1[(1992), pp.]: 277-298, 1992) provides an optimal solution.

Amend the paragraph beginning at column 37, line 25, as follows:

If option (a) immediately above is chosen, then the auxiliary functions are used as approximation functions for obtaining at least a near-optimal solution to the optimization problem of step 508. Note that the auxiliary functions used depend on the value of m. Thus, auxiliary functions used when $m=3$ will likely be different from those for $m=4$. But in any case, the optimization procedure is guided by using the merit function, $\Phi_{2n}(u^3,\ldots,u^n)$, which can be computed exactly via a [2-dimensional] two-dimensional assignment problem for guiding the maximization process.

Amend the paragraph beginning at column 37, line 34, as follows:

Alternatively, if option (b) above is chosen, then two important advantages result[:], namely, the optimization problem of step 508 can be always solved optimally and without using auxiliary approximation functions. Thus, better solutions to the original M-dimensional assignment problem are likely since there is no guarantee that when the non-smooth optimization techniques are applied to the auxiliary functions the techniques will yield an optimal solution to step 508. Furthermore, it is important to note that without auxiliary functions, the processing in step 508 is both conceptually easier to follow and more efficient.

Amend the paragraph beginning at column 37, line 45, as follows:

Subsequently, in step 512 of Fig. 8, the solution $(\overline{u}^{1\ ]m+1},\ldots,\overline{u}^{n})$ [and w^n]is used in determining an optimal

solution[,] w^m to Problem Formulation [[6.4]]<u>(2.9)</u> as generated according to [[6.3]]<u>(2.8)</u> and Fact A.3.

Amend the paragraph beginning at column 37, line 49, as follows:

In step 516, the solution w^n is used in defining the cost matrix c^{n-m+1} as in [[6.7]]<u>(2.12)</u>. Subsequently, if $n-m+1=2$, then the assignment problem [[6.8]]<u>(2.13)</u> may be solved straightforwardly using known techniques such as forward/reverse auction algorithms. Following this, in step 528, the solution to the [2-dimensional]<u>two-dimensional</u> assignment problem is assigned to the variable ASSIGNMT_SOLUTION and in step 532 ASSIGNMT_SOLUTION is returned to a dimension three level recursion of MULTI_DIM_RELAX for solving a [3-dimensional]<u>three-dimensional</u> assignment problem.

Amend the paragraph beginning at column 37, line 59, as follows:

Alternatively, if in step 520, $n-m+1>2$, then in step 536 the data structure(s) representing a problem formulation as in [[6.8]]<u>(2.13)</u> is generated and assigned to the parameter, PROB_FORMULATION. Subsequently, in step 540 a recursive copy of MULTI_DIM_RELAX is invoked to solve the lower dimensional assignment problem represented by PROB_FORMULATION. Upon the completion of step 540, the parameter, ASSIGNMT_SOLUTION, contains the solution to the [n-m+1 dimensional]<u>$(n-m+1)$-dimensional</u> assignment problem. Thus, in step 544, the [n-m+1]<u>$(n-m+1)^{th}$</u> solution is used to solve the n-dimensional assignment problem as discussed regarding equations [[6.9]]<u>(2.14)</u>. Finally, in steps 548 and 552 the solution to the n-dimensional assignment problem is returned to the calling program so that, for example, it may be used in taking one or more actions such as (a) sending a warning to aircraft or sea

facility; (b) controlling air traffic; (c) controlling anti-aircraft or anti-missile equipment; (d) taking evasive action; or (e) surveilling an object.

Amend the paragraph beginning at column 38, line 12, as follows:

There are many procedures described by Figure 8. One such procedure is the first embodiment of the]multi dimensional] multi-dimensional assignment solving process of section I.1. That is, by specifying $m=n-1$ in step 500, a single set of constraints is relaxed in step 508. Thus, one set of constraints is incorporated[is] into the objective function via the Lagrangian problem formulation, resulting in an [m=n-1 dimensional] $(m=n-1)$-dimensional problem. The relaxed problem is subsequently maximized in step 512 with respect to the corresponding Lagrange multipliers and then a feasible solution is reconstructed for the n-dimensional problem using a two-dimensional assignment problem. The second procedure provided by Figure 8 is a novel approach which is not suggested by the first embodiment of section I.1. In fact, the second procedure is somewhat of a mirror image of the first embodiment in that n-2 sets of constraints are simultaneously relaxed, yielding immediately an [m=2 dimensional] $(m=2)$-dimensional problem in step 512. Thus, a feasible solution to the n-dimensional problem is then recovered using a recursively obtained solution to an [n-1 dimensional] $(n-1)$-dimensional problem via step 540. In this case, the function values and subgradients of $\Phi_{2n}(u^3,...,u^n)$ of step 508 can be computed optimally via a [two dimensional] two-dimensional assignment problem. The significant advantage here is that there is no need for the merit or auxiliary functions, Ψ_k, as required in the first embodiment of the [multidimensional] multi-dimensional assignment solving process of section I.1 above and also there is no need for the more

general merit or auxiliary functions Ψ_{mn}, as discussed in subsection II.4.2 below. Further, note that all function values and subgradients used in the [non-smooth] nonsmooth maximization process are computed exactly (i.e., optimally) in this second procedure. Moreover, problem decomposition is now carried out for the n-dimensional problem; however, decomposition of the [n-1 dimensional] (n-1)-dimensional recovery problem (and all lower recovery problems) is performed only after the problem is formulated. Between these two procedures are a host of different relaxation schemes based on relaxing n - m sets of constraints to an m-dimensional problem ([2<m<n]$2<m<n$), but these all have the same difficulties as the procedure for the first embodiment of section I.1 in that the relaxed problem is an NP-hard problem. To resolve this difficulty, we use an auxiliary or merit function Ψ_{mn} as described in subsection II.4.2 below. (The notation Ψ_k was used for $\Psi_{n-1,n}$ in section I.1.) For the case $m<n-1$, the recovery procedure is still based on an NP-hard ([n-m+1-dimensional]$n-m+1$)-dimensional assignment problem. Note that the partitioning techniques similar to those discussed in Section I.3 may be used to identify the assignment problem with a layered graph and then to find the disjoint components of this graph. In general, all relaxed problems can be decomposed prior to any [non-smooth] nonsmooth computations because their structure stays fixed throughout the algorithm of Fig. 8. However, all recovery problems cannot be decomposed until they are formulated, as their structure changes as the solutions to the relaxed problems change.

Amend the paragraph beginning at column 39, line 4, as follows:

One of the key steps in the Lagrangian relaxation algorithm in section II.3 is the solution of the problem

[[7.1]](2.15)

[Maximize $\{\Phi_{mn}(u^{m+1},...,u^n) : u^k \in \mathbb{R}^{M_k+1};\ k=m+1,...,n\}$]

$\underline{\textit{Maximize } \{\Phi_{mn}(u^{m+1},...,u^n) : u^k \in \mathbb{R}^{M_k+1};\ k=m+1,...,n\},}$

where $u_0^k=0$ for all $k=m+1,...,n$. The evaluation of $\Phi_{mn}(u^{m+1},...,u^n)$ requires the optimal solution of the corresponding minimization problem [[6.2] Z^n varies for each instance of $(u^{m+1},...,u^n)$]$\underline{(2.7)}$. The following discussion provides some properties of these functions.

Amend the paragraph beginning at column 39, line 12, as follows:

Fact A.4. Let $u^{m+1},...,u^n$ be multiplier vectors associated with the $(m+1)^{st}$ through the n^{th} set of constraints [[6.1]]$\underline{(2.6)}$, let Φ_{mn} be as defined in [[6.1]]$\underline{(2.6)}$, let $V_n(\underline{\bar{z}}^{l\ ]n})$ be the objective function value of the n-dimensional assignment problem in equation [[6.1]]$\underline{(2.6)}$, let z^n be any feasible solution of [[6.1]]$\underline{(2.6)}$, and let $\underline{\bar{z}}^{l\ ]n}$ be an optimal solution of [[6.1]]$\underline{(2.6)}$. Then, [$\Phi_{mn}(u^{m+1},...,u^n)$] $\Phi_{mn}(u^{m+1},...,u^n)$ is piecewise affine, concave and continuous in $\{u^{m+1},...,u^n\}$ and

$$\Phi_{mn}(u^{m+1},...,u^n) \leq V_n(\underline{\bar{z}}^n) \leq V_n(z^n)\underline{.} \qquad [[7.2a]]\underline{(2.16})$$

Furthermore,

$$\Phi_{m-1n}(u^m,u^{m+1},...,u^n) \leq \Phi_{mn}(u^{m+1},...,u^n) \qquad [[7.2a]]\underline{(2.17})$$

for all $u^k \in \mathbb{R}^{M_k+1}$ with $u_0^k=0$ and $k=m,...,n$.

Amend the paragraph beginning at column 39, line 23, as follows:

Most of the procedures for non-smooth optimization are based on generalized gradients called subgradients, given by the following definition.

Definition. *At $u=(u^{m+1},...,u^n)$ the set* [$\delta\Phi_{mn}(u)$] $\underline{\partial\Phi_{mn}(u)}$ *is called a subdifferential of Φ_{mn}* $\underline{\textit{and defined by}}$

[$\delta\Phi_{mn}(u)=\{q\in\mathbb{R}^{M_{m+1}+1}\text{x}...\text{x}\mathbb{R}^{M_n+1} \mid \Phi_{mn}(w)-\Phi_{mn}(u)\leq q^T(w-u)\ \forall w\in\mathbb{R}^{M_{m+1}+1}\text{x}...\text{x}\mathbb{R}^{M_n+1}\}$

[7.3]]

$\underline{\partial\Phi_{mn}(u)=\{g\in\mathbb{R}^{M_{m+1}+1}\times\ldots\times\mathbb{R}^{M_n+1}\,|\,\Phi_{mn}(w)-\Phi_{mn}(u)\leq g^T([w-u)\,\forall w\in\mathbb{R}^{Mm+1+1}x\ldots x\mathbb{R}^{Mn+1}}$ (2.18)

[A vector $q\in\delta\Phi_{mn}\gamma$]A vector $q\in\delta\Phi_{mn}(u)$ is called a subgradient.

Amend the paragraph beginning at column 39, line 32, as follows:

If z^n is an optimal solution of [[6.2]](2.7) computed during evaluation of [$\Phi_{mn}\gamma$ $\Phi_{mn} i_n$ ^{n}] $\Phi_{mn}(u)$, differentiating Φ_{mn} with respect to $u^n_{i_n}$ yields the following [i_n-th]i_n^{th} component of a subgradient g of [$\Phi_{mn}\gamma$] $\Phi_{mn}(u)$

$$g_0^k=0$$

$$g_{i_k}^k=\sum_{i_1=0}^{N_1}\cdots\sum_{i_{k-1}=0}^{N_{k-1}}\sum_{i_{k+1}=0}^{N_{k+1}}\cdots\sum_{i_n=0}^{N_n} z^n_{i_1\ldots i_n}-1$$

[[7.4]](2.19)

for $i_k=1,\ldots,N_k$ and $k=m+1,\ldots,n$.

If z^n is the unique optimal solution of [[6.2]](2.7), [$\delta\Phi_{mn}\gamma$ Φ_{mn}] $\partial\Phi_{mn}(u)=\{g\}$ and Φ_{mn} is differentiable at u. If the optimal solution is not unique, then there are finitely many such solutions, say $z^n(1),\ldots,z^n(K)$. Given the corresponding subgradients, $g^1,\ldots,g^K$, the subdifferential [$\delta\Phi\gamma$]$\partial\Phi(u)$ is the convex hull of $\{g^1,\ldots,g^K\}$.

Amend the paragraph beginning at column 39, line 51, as follows:

For real-time needs, one must address the fact that the non-smooth optimization problem of step 508 (Fig. 8) requires the solution of an NP-hard problem for m>2. One approach to this problem is to use the following merit or auxiliary function to decide whether a function value has increased or decreased sufficiently in the line search or trust region methods:

[[7.5]]<u>(2.20)</u>

$$\Psi_{mn}(\bar{u}^3,\ldots,\bar{u}^m;u^{m+1},\ldots,u^n) =$$

$$\begin{cases} \Phi_{mn}(u^{m+1},\ldots,u^n) & \text{if } m = 2 \\ & \text{or [(6.2)](2.7) is solved optimally,} \\ \Phi_{2n}(\bar{u}^3,\ldots,\bar{u}^m;u^{m+1},\ldots,u^n) & \text{otherwise.} \end{cases}$$

where the multipliers $\bar{u}^3,\ldots,\bar{u}^m$ that appear in lower order relaxations used to construct (suboptimal) solutions of the m-dimensional relaxed problem [(6.2)]<u>(2.7)</u> have been explicitly included. Note that [Ψ_{mn}Y] <u>Ψ_{mn} is well-defined since (2.7) can always</u> be solved optimally if m=2. For sufficiently small problems [(6.2)]<u>(2.7)</u> or [(6.4)]<u>(2.9)</u>, one can more efficiently solve the NP-hard problem by branch and bound. This is the reason for the inclusion of the first case; otherwise, the relaxed function [Φ_{2n}] <u>Φ_{2n} is used to guide the nonsmooth optimization phase. That</u> the merit function provides a lower bound for the optimal solution follows directly from Fact A.4 and the following fact.

Amend the paragraph beginning at column 40, line 9, as follows:

Fact A.5. Given the definition of [Ψ_{mn}Y] <u>Ψ_{mn} in (2.20) above,</u>

$\Psi_{mn}(\bar{u}^3,\ldots,\bar{u}^m;u^{m+1},\ldots,u^n) \le \Phi_{mn}(u^{m+1},\ldots,u^n)$ for all multipliers $\bar{u}^3,\ldots,\bar{u}^m,u^{m+1},\ldots,u^n$. [[7.6]]<u>(2.21)</u>

The actual function value used in the optimization phase is [Ψ_{mn}Y] <u>Ψ_{mn}; however, the subgradients are computed as in (2.19), but</u> with the solution $z^n_{i_1\ldots i_n}$ being a suboptimal solution constructed from a relaxation procedure applied to the m-dimensional problem. Again, the use of these lower order relaxed problems is the reason for the inclusion of the multipliers $\bar{u}^3,\ldots,\bar{u}^m$.

Delete the paragraph beginning at column 40, line 18, and replace it with the following new paragraph:

To explain how the merit function is used, suppose there is a current multiplier set $(u^{m+1}_{old}, \ldots, u^{n}_{old})$ and it is desirable to update to a new multiplier set $(u^{m+1}_{new}, \ldots, u^{n}_{new})$ via $(u^{m+1}_{new}, \ldots, u^{n}_{new}) = (u^{m+1}_{old}, \ldots, u^{n}_{old}) + (\Delta u^{m+1}, \ldots, \Delta u^{n})$. Then $\Psi_{mn}(\bar{u}^{3}_{old}, \ldots, \bar{u}^{m}_{old};\ u^{m+1}_{old}, \ldots, u^{n}_{old})$ is computed where $(\bar{u}^{3}_{old}, \ldots, \bar{u}^{m}_{old})$ is obtained during the relaxation process used to compute a high quality solution to the relaxed m-dimensional assignment problem(2.7) at $(u^{m+1}, \ldots, u^{n}) = (u^{m+1}_{old}, \ldots, u^{n}_{old})$. The decision to accept $(u^{m+1}_{new}, \ldots, u^{n}_{new})$ is then based on $\Psi_{mn}(\bar{u}^{3}_{old}, \ldots, \bar{u}^{m}_{old};\ u^{m+1}_{old}, \ldots, u^{n}_{old}) < \Psi_{mn}(\bar{u}^{3}_{new}, \ldots, \bar{u}^{m}_{new};\ u^{m+1}_{new}, \ldots, u^{n}_{new})$ or some other stopping criteria commonly used in line searches. Again, $(\bar{u}^{3}_{new}, \ldots, \bar{u}^{m}_{new})$ represents the multiplier set used in the lower level relaxation procedure to construct a high quality feasible solution to the m-dimensional relaxed problem (2.7) at $(u^{m+1}, \ldots, u^{n}) = (u^{m+1}_{new}, \ldots, u^{n}_{new})$. The point is that each time one changes $(u^{m+1}, \ldots, u^{n})$ and uses the merit function $\Psi_{mn}(\bar{u}^{3}, \ldots, \bar{u}^{m};\ u^{m+1}, \ldots, u^{n})$ for comparison purposes, one must generally change the lower level multipliers $(\bar{u}^{3}, \ldots, \bar{u}^{m})$.

Amend the paragraph beginning at column 40, line 36, as follows:

An illustration of this merit function for [m=n-1] $m=n-1$ is given in ["PARTITIONING MULTIPLE DATA SETS, MULTIDIMENSIONAL ASSIGNMENTS AND LAGRANGIAN RELAXATION", by Aubrey B. Poore and Nenad Rijavec, and in "QUADRATIC ASSIGNMENT AND RELATED PROBLEMS, DIMACS SERIES IN DISCRETE MATHEMATICS AND THEORETICAL COMPUTER SCIENCE", in American Mathematical Society, Providence, R.I., Vol. 16, 1994, pp. 25-37 edited by Panos M. Pardalos and Henry Wolkowicz] (A.B. Poore and N. Rijavec. Partitioning Multiple Data Sets, Multidimensional Assignments and Lagrangian Relaxation. In

P.M. Pardalos and H. Wolkowicz, editors, Quadratic Assignment and Related Problems: DIMACS Series in Discrete Mathematics and Theoretical Computer Science, volume 16, pages 25-37, 1994).

Amend the paragraph beginning at column 40, line 47, as follows:

By Fact A.4 the function [$\Phi_{mn}\gamma$] $\Phi_{mn}(u)$ is a continuous, piecewise affine, and concave, so that the negative of [$\Phi_{mn}\gamma\Delta$] $\Phi_{mn}(u)$ is convex. Thus the problem of maximizing [$\Phi_{mn}\gamma\Delta$] $\Phi_{mn}(u)$ is one of nonsmooth optimization. Since there is a large literature on such problems, only a brief discussion of the primary classes of methods for solving such problems is provided. A first class of methods, known as subgradient methods, are reviewed and analyzed in [the book by](N.Z. Shor[,]. *Minimization Methods for Non-Differentiable Functions*[,]. Springer-Verlag, New York, 1985). Despite their relative simplicity, subgradient methods have some drawbacks that make them inappropriate for the tracking problem. They do not guarantee a descent direction at each iteration, they lack a clear stopping criterion, and exhibit poor convergence (less than linear).

Amend the paragraph beginning at column 40, line 60, as follows:

A more recent and sophisticated class of methods are the bundle methods; excellent developments are presented in [the books of Hiriart-Urruty and Lemaréchal:](J.-B. Hiriart-Urruty and C. Lemaréchal[,]. *Convex Analysis and Minimization Algorithms I* and *II*[,]. Springer-Verlag, Berlin, 1993; [and the book of]K.C. Kiwiel[:]. Methods of Descent for Non-Differentiable Optimization. In A. Dold and B. Eckmann, editors, *Lecture Notes in Mathematics*, [Vol.]volume 1133, Berlin, 1985. Springer-Verlag[, Berlin, 1985]). Bundle methods retain a set (or bundle) of previously computed subgradients to determine the best

possible descent direction at the current iteration. Because of the [non-smoothness] nonsmoothness of Φ_{mn}, the resulting direction may not provide a "sufficient" decrease in Φ_{mn}. In this case, bundle algorithms take a "null" step, wherein the bundle is enriched by a subgradient close to u_k. As a result, bundle methods are non-ascent methods which satisfy the relaxed descent condition [$\Phi_{mn}(u_{k+1})<\Phi_{mn}(u_k)$ if $u_{k+1}\neq u_k$] $\Phi_{mn}(u_{k+1})<\Phi_{mn}(u_k)$ if $u_{k+1}\neq u_k$. These methods have been shown to have good convergence properties. In particular, bundle method variants have been proven to converge in a finite number of steps for piecewise affine convex functionals in (K.C. Kiwiel[,]. An [Aggregate Subgradient Method for Non-smooth Convex Minimization] aggregate subgradient method for non-smooth convex minimization[,]. *Mathematical Programming*, 27[, (1983) pp.]:320-341, [and]1983; H. Schramm and J. Zowe[,]. A version of the bundle idea for minimizing a non-smooth function: Conceptual idea, convergence analysis, numerical results[,]. *SIAM Journal on Optimization*, 2[(1992)], [pp]No. 1:121-152, February, 1992).

Amend the paragraph beginning at column 41, line 19, as follows:

All of the above non-smooth optimization methods require the computation [$\Phi_{mn}\gamma\Delta\in\delta\Phi_{mn}\gamma\Delta$] of $\Phi_{mn}(u)$ and a $g \in \partial\Phi_{mn}(u)$. These in turn require the computation of the relaxed cost coefficients [[6.3]](2.8). In both the first and second procedures discussed in section II.3.1, maximization of [$\Phi_{2n}\Lambda$] $\Phi_{2n}(u)$ must be repeatedly evaluated. In the most efficient implementations presently known of these two procedures, it was found that at least 95% of the computational effort of the entire procedure is spent in the evaluation of the relaxed cost coefficients [[6.3]](2.8) as part of computing [$\Phi_{2n}\gamma\Lambda$] $\Phi_{2n}(u)$. Thus, generally

a method should be chosen that makes as efficient use of the subgradients and function values as [is]possible, even at the cost sophisticated manipulation of the subgradients. In evaluating three different bundle procedures: (a) the conjugate subgradient method of Wolfe used in section I.1 of the first embodiment of the present invention; (b) the aggregate subgradient method of Kiwiel ([described in]K.C. Kiwiel[,]. An [Aggregate Subgradient Method for Non-smooth Convex Minimization] aggregate subgradient method for non-smooth convex minimization[,]. *Mathematical Programming*, 27[, (1983) pp.]:320-341, 1983); and (c) the bundle trust method of Schramm and Zowe ([described in]H. Schramm and J. Zowe[,]. A version of the bundle idea for minimizing a non-smooth function: Conceptual idea, convergence analysis, numerical results[,]. *SIAM Journal on Optimization*, 2[(1992)], [pp]No. 1:121-152, February, 1992), it was determined that for a fixed number of non-smooth iterations, say, ten, the bundle-trust method provides good quality solutions with the fewest number of function and subgradient evaluations of all the methods, and is therefore the preferred method.

Amend the paragraph beginning at column 41, line 47, as follows:

The forward/reverse auction algorithm of Bertsekas and Castañon ([as described in]D.P. Bertsekas and D.A. Castañon[,]. A forward/reverse auction algorithm for asymmetric assignment problems[,]. *Computational Optimization and Applications*, 1 [(1992), pp.]:277-298, 1992) is used to solve the many relaxed [two dimensional] two-dimensional problems that occur in the course of execution.

Amend the paragraph beginning at column 41, line 55, as follows:

The effective use of "hot starts" is fundamental for real-time applications. A good initial set of multipliers can significantly reduce the number of non-smooth iterations (and hence the number of [Φ_{2n}γΔ] Φ_{mn} evaluations) required for a high quality recovered solution. Further, there are several ways that multipliers can be reused. First, if [a] an n-dimensional problem is relaxed to [a] an m-dimensional problem, relaxation provides the multiplier set $\{u^{m+1}, u^{m+2},\ldots,u^{n}\}$. These can be used as the initial multipliers for the [n-m+1 dimensional] $(n-m+1)$-dimensional recovery problem for $n-m+1>2$. This approach has also worked well to reduce the number of non-smooth iterations during recovery.

Amend the paragraph beginning at column 41, line 66, as follows:

Further, for track maintenance, initial feasible solutions are generated as follows. When a new scan of information (a new observation set) arrives from a sensor, one can obtain an initial primal feasible solution by matching new reports to existing tracks via a [two dimensional] two-dimensional assignment problem. This is known as the track-while-scan (TWS) approach. Thus, an initial primal solution exists and then we use the above relaxation procedure [is]to make improvements to this TWS solution. Also for track maintenance, multipliers are available from a previously solved and closely related [multi-dimensional] multidimensional assignment problems for all but the new observation set.

Amend the paragraph beginning at column 42, line 12, as follows:

Given a feasible solution of the [multi-dimensional] multidimensional assignment problem, one can consider local search procedures to improve this result, as described in (C.H. Papadimitriou and K. Steiglitz[,]. *Combinatorial Optimization: Algorithms and Complexity*[,]. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1982[, and]; J. Pearl[,]. *Heuristics: [intelligent search strategies for computer problem solving,] Intelligent Search Strategies for Computer Problem Solving.* Addison-Wesley, Reading, MA, 1984). One method is the idea of [k-interchanges] *k* interchanges. Since for sparse problems only those active arcs that participate in the solution are stored, it is difficult to efficiently identify eligible variable exchange sets with some data structures for solving the assignment problem. However, a local search that may be more promising is to use the [two dimensional] two-dimensional assignment solver in the following way. Given a feasible solution to the [multi-dimensional] multidimensional assignment problem, the indices that correspond to active arcs in the solution are enumerated. Subsequently, one coordinate is freed to formulate a [two dimensional] two-dimensional assignment problem with one index corresponding to the enumeration and the other to the freed coordinate, and then solve a [two dimensional] two-dimensional assignment problem to improve the freed index position.

Amend the paragraph beginning at column 42, line 34, as follows:

The procedures described thus far are all based on relaxation. Due to the sparsity of assignment problems, however, frequently a decomposition of the problem into a collection of disjoint components can be done wherein each of the components

can be solved independently. Due to the setup costs of Lagrangian relaxation, a branch and bound procedure is generally more efficient for small components, say ten to twenty feasible 0-1 variables (i.e., $z_{i_1 \ldots i_n}$). Otherwise, the relaxation procedures described above are used. Perhaps the easiest way to view the decomposition method is to view the reports or measurements as a layered graph. A vertex is associated with each observation point, and an edge is allowed to connect two vertices only if the two observations belong to at least one feasible track of observations. Given this graph, the decomposition problem can then be posed as that of identifying the connected subcomponents of a graph which can be accomplished by constructing a spanning forest via a depth first search algorithm, as described in (A.V. Aho, J.E. Hopcroft, and J.D. Ullman[,]. *The Design and Analysis of Computer Algorithms*[,]. Addison-Wesley[Publishing Company, Reading], MA, 1974). Additionally, decomposition of a different type might be based on the identification of bi- and tri-connected components of a graph and enumerating on the connections. Here is a technical explanation. Let

[$\mathcal{Z} = \{z_{i_1 i_2 \ldots i_n} | z_{i_1 i_2 \ldots i_n}$ is not preassigned to zero$\}$]

$\mathcal{Z} = \{z_{i_1 i_2 \ldots i_n} | z_{i_1 i_2 \ldots i_n}$ *is not preassigned to zero*$\}$

denote the set of assignable variables. Define an undirected graph G(N,A) where the set of nodes is

[$N=\{z^n_{i_n} | n=1, \ldots, n;\ i_n=1, \ldots, N_m\}$]

$N=\{z^n_{i_n} | n=1, \ldots, n;\ i_n=1, \ldots, N_m\}$

and arcs,

[$A=\{(z^n_{j_n}, z^1_{j_1}) | n \neq 1]$, $j_n \neq 0$, $j_1 \neq 0$ and there exists $z_{i_1 i_2 \ldots i_n} \in \mathcal{Z}$ such that $j_n=i_n$ and $j_1=i_1\}$]

$A=\{(z^n_{j_n}, z^l_{j_l})$ [$|n \neq 1$] $|n \neq 1,\ j_n \neq 0,\ j_l \neq 0$ *and there exists* $z_{i_1 i_2 \ldots i_n} \in \mathcal{Z}$.

Amend the paragraph beginning at column 43, line 2, as follows:

Note that the nodes corresponding to zero index have not been included in the above defined graph, since two variables that have only the zero index in common can be assigned independently. Connected components of the graph are then easily found by constructing a spanning forest via a depth first search. (A detailed algorithm can be found in the book by Aho, Hopcroft and Ullman cited above). Furthermore, this procedure is used at each level in the relaxation, i.e., is applied to each assignment problem [[3.1]]<u>(1.4)</u> for n=3,...,M.

Amend the paragraph beginning at column 43, line 12, as follows:

The original relaxation problem is decomposed first. All relaxed assignment problems can be decomposed <u>a priori</u> and all recovery problems can be decomposed only after they are formulated. Hence, in the n-to-(n-1) case, we have n-2 relaxed problems that can all be decomposed initially, and the recovery problems that are not decomposed (since they are all [two dimensional] <u>two-dimensional</u>). In the n-to-2 case, we have only one relaxed problem that can be decomposed initially. This case yields n-3 recovery problems, which can be decomposed only after they are formulated.

Insert the following heading before the paragraph beginning at column 43, line 22:

<u>III. New Approaches to Track Initiation and Maintenance Using Multidimensional Assignment Problems</u>

Amend the paragraph beginning at column 43, line 22, as follows:

The ever-increasing demand for sensor surveillance systems is to accurately track and identify objects in real-time, even for dense target scenarios and in regions of high track contention. The use of multiple sensors, through more varied

information, has the potential to greatly improve target state estimation and identification. However, to take full advantage of the available data, a multiple frame data association approach is needed. The most popular such approach is an enumerative technique called multiple hypothesis tracking (MHT). As an enumerative technique, MHT can be too computationally intensive for real-time needs because this problem is NP-hard. A promising approach is to utilize the recently developed Lagrangian relaxation algorithms (K.P.Pattipati, S. Deb, and Y.Bar-Shalom. A multisensor-multitarget data association algorithm for heterogeneous sensors. *IEEE Transactions on Aerospace and Electronic Systems*, 29, No. 2:560-568, April 1993; A.B. Poore and N.Rijavec. Partitioning multiple data sets: multidimensional assignments and lagrangian relaxation, in quadratic assignment and related problems, P.M. Pardalos and H.Wolkowicz, editors, *DIMACS series in Discrete Mathematics and Theoretical Computer Science*, 16:25-37, 1994; A.J. Robertson III. A class of lagrangian relaxation algorithms for the multidimensional problem. *Ph.D. Thesis, Colorado State University, Ft. Collins, CO, 1995)* for the multidimensional assignment problem; however, there are many other potentially good approaches to these assignment problems such as LP relaxation combined with an interior point method, GRASP, and parallelization.

Amend the paragraph beginning at column 43, line 57,as follows:

III.2. Overview of the Tracking Problem

Amend the paragraph beginning at column 43, line 58, as follows:

Tracking and data fusion are complex subjects of specialization that can only be briefly summarized as they are related to the subject of this paper. The processing of track multiple targets using data from one or more sensors is typically

partitioned into two stages or major functional blocks, namely, signal processing and data [process]processing (Y.Bar-Shalom. *Multitarget-Multisensor Tracking: Advanced* Applications. Artech House, Dedham, MA, 1990; Y.Bar-Shalom and T.E. Fortmann. *Tracking and Data Association.* Academic Press, Boston, MA, 1988; S.S. Blackman. *Multiple Target Tracking with Radar Applications.* Artech House, Dedham, MA, 1986). The first stage is the signal processing that takes raw sensor data and outputs reports. Reports are sometimes called observations, threshold exceedances, plots, hits, or returns, depending on the type of sensor. The true source of each report is usually unknown and can be due to a target of interest, a false signal, or persistent background objects that can be moving in the field of view of the sensor.

Amend the paragraph beginning at column 44, line 23, as follows:

III.2.1 Tracking Functions

Amend the paragraph beginning at column 44, line 36, as follows:

In track maintenance, the data association function decides how to relate the reports from the current frame of data to the previously computed tracks. In one approach, at most one report is assigned to each track, and in other approaches, weights are assigned to the pairings of reports to a track. After the data association, the filter function updates each target state estimate using the one or more (with weights) reports that were determined by the data association function. A filter commonly used for multiple target tracking and data fusion is the well known Kalman filter (or the extended Kalman filter in the case of nonlinear problems) or a simplified version of it (Y. Bar-Shalom. *Multitarget-Multisensor Tracking: Advanced Applications.* Artech House, MA, 1990; Y.Bar-Shalom and T.E. Fortmann. *Tracking and*

Data Association. Academic Press, Boston, MA, 1988; S.S. Blackman. *Multiple Target Tracking with Radar Applications.* Artech House, Dedham, MA, 1986).

Amend the paragraph beginning at column 44, line 62, as follows:

There have been numerous approaches developed to perform the data association function. Since optimal data association is far too complex to implement, good but practical sub-optimal approaches are pursued. Data association approaches can be classified in a number of ways. One way to classify data association approaches is based on the number of data frames used in the association process[.] (O.E. Drummond. Multiple sensor tracking with multiple frame, probabilistic data association. In *Signal and Data Processing of Small Targets, SPIE Proceedings,* volume 2561, pages 322-336, 1995). In single frame data association for track maintenance, "hard decisions" are made [each frame as to how the]on the assignment of reports[are to be related] to the tracks. Some single frame approaches include: individual nearest neighbor, PDA, JPDA, and global nearest neighbor (sequential most probable hypothesis tracking), which uses a [two dimensional] two-dimensional assignment algorithm (Y. Bar-Shalom. *Multitarget-Multisensor Tracking: Advanced Applications.* Artech House, MA, 1990; Y. Bar-Shalom and T.E. Fortmann. *Tracking and Data Association.* Academic Press, Boston, MA, 1988; S.S. Blackman. *Multiple Target Tracking with Radar Applications.* Artech House, Dedham, MA, 1986). In most single frame data association approaches, only one track per object is carried forward to be used in processing the next frame of reports.

Amend the paragraph beginning at column 45, line 11, as follows:

Multiple frame data association is more complex and frequently involves purposely [carry]carrying forward more than one track per target to be used in processing the next frame of reports. By retaining more than one track per target, the tracking performance is improved at the cost of increased processing load. The best known multiple frame data association approach is an enumerative technique called multiple hypothesis tracking (MHT) which enumerates all the global hypothesis with various pruning rules.

Amend the paragraph beginning at column 45, line 59, as follows:

The integrated approach just discussed, uses the same number of frames of reports for both track initiation and track maintenance. The goal in using the multi-dimensional assignment algorithm is to provide improved performance while minimizing the amount of processing required. Typically, track initiation will benefit from more frames of reports than will track maintenance. Thus, a second approach that integrates track initiation and track maintenance is discussed hereinafter, wherein the number of frames of reports is not the same for these two functions. This is a novel approach that is introduced in this paper.

Amend the paragraph beginning at column 46, line 4, as follows:

There are many advantages and many ways [to]of combining data from multiple sensors. There are also many ways of categorizing the different algorithmic architectures for processing data from multiple sensors. One approach outlines four generic algorithmic architectures (O.E. Drummond. Multiple sensor tracking with multiple frame, probabilistic data

association. In *Signal and Data Processing of Small Targets, SPIE Proceedings*, volume 2561, pages 322-336, 1995). Two of these generic architectures are especially pertinent to this paper and are summarized briefly.

Amend the paragraph beginning at column 46, line 11, as follows:

In the Centralized Fusion algorithmic architecture, reports are combined from the various sensors to form global tracks. This algorithmic architecture is also called Central Level Tracking, Centralized Algorithmic Architecture, or simply the Type IV algorithmic architecture. In track maintenance, for example, if a single frame data association is used, then for data association a frame of reports from one sensor is processed with the latest global tracks; then the global tracks are updated by the filter function. After the processing of this frame of reports is completed, a frame of the reports from another (or the same) sensor is processed with these updated global tracks. This process is continued as new frames of data become available to the system as a whole.

Amend the paragraph beginning at column 46, line 35, as follows:

The second pertinent algorithmic architecture is Track Fusion. This approach is also called the Hierarchical or Federated algorithmic architecture, sensor level tracking or simply the Type II algorithmic architecture. In track maintenance, for example, a processor for each sensor computes single sensor tracks; these tracks are then forwarded to the global tracker to compute global tracks based on data from all sensors.

Amend the paragraph beginning at column 46, line 47, as follows:

After the first time a sensor processor forwards tracks to the global tracker, then subsequent tracks for the same targets are cross-correlated with the existing global tracks. This track-to-track cross-correlation is due to the common history of the current sensor tracks and the tracks from the same sensor that were forwarded earlier to the global tracker. The processing must take this cross-correlation into account and there are a number of ways of compensating for this-cross-correlations. One method for dealing with this cross-correlation is to decorrelate the sensor tracks that are sent to the global tracker. There are a variety of ways to achieve this decorrelation and some are summarized in [Drummond, 1996, *Signal Data Proc.*] <u>recent paper (O.E. Drummond, Feedback in track fusion without process noise. In *Signal and Data Processing* of Small Targets</u> *[1995]*<u>, *SPIE Proceedings*</u>*,[-2561:]* <u>volume 2561, pages</u> 369-383<u>, 1995</u>.

Amend the paragraph beginning at column 47, line 2, as follows:

<u>III.3</u> Formulation of the Data Association Problem

Amend the paragraph beginning at column 47, line 4, as follows:

The goal of this section is to briefly outline the probabilistic framework for the data association problems presented in this work. The technical details are presented elsewhere <u>(A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994)</u>. The data association problems for multisensor and multitarget tracking considered in this work are

generally posed (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994) as that of maximizing the posterior probability of the surveillance region (given the data) according to

$$\textit{Maximize}\ \{P(\Gamma=\gamma \mid Z^{N+1}) \mid \gamma \in \Gamma^*\} \qquad [(3.1)]$$

where Z^{N+1} represents N+1 data sets, γ is a partition of indices of the data (and thus induces a partition of the data), Γ^* is the finite collection of all such partitions, Γ is a discrete random element defined on Γ^*, [γ^0 is a reference partition,] and $P(\Gamma=\gamma|Z^{N+1})$ is the posterior probability of a partition γ being true given the data Z^{N+1}. The term partition is defined below; however, this framework is currently sufficiently general to cover set packings and coverings.

Amend the paragraph beginning at column 47, line 27, as follows:

Consider N+1 data sets Z(k)(k=1,...,N+1), each consisting of M_k actual reports and a dummy report z_0^k, and let denote the cumulative data set defined by

$$Z(k) = \left\{z_{i_k}^k\right\}_{i_k=0}^{M_k} \text{ and } Z^{N+1}=\{Z(1),...,Z(N+1)\}, \qquad (3.2)$$

respectively. (The dummy report z_0^k serves several purposes in the representation of missing data, false reports, initiating tracks, and terminating tracks (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational*

Optimization and Applications, 3:27-57, 1994).) In multisensor data fusion and multitarget tracking the data sets Z(k) may represent different classes of objects, and each data set can arise from different sensors. For track initiation the objects are reports that must be partitioned into tracks and false alarms. In our formulation of track maintenance, which uses a moving window, one data set will be tracks and remaining data sets will be reports which are assigned to existing tracks, as false reports, or to initiating tracks.

Amend the paragraph beginning at column 47, line 44, as follows:

We specialize the problem to the case of set partitioning (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications,* 3:27-57, 1994) defined in the following way. Define a "track of data" as where each i_k [and] can assume [the] zero or nonzero values[of 0 and , respectively]. A partition of the data will refer to a collection of tracks of data wherein each report occurs exactly once in one of the tracks of data and such that all data is used up; the occurrence of dummy report is unrestricted. The reference partition γ° is that in which all reports are declared to be false.

Amend the paragraph beginning at column 47, line 53, as follows:

Next, under appropriate independence assumptions one can show

$$\frac{P(\Gamma = \gamma \mid Z^{N+1})}{P(\Gamma = \gamma^0 \mid Z^{N+1})} \equiv L\gamma \equiv \prod_{(i_1 \cdots i_{N+1}) \in \gamma} L_{i_1 \cdots i_{N+1}}, \tag{3.3}$$

$L_{i_1 \cdots i_{N+1}}$ is a likelihood ratio containing probabilities for detection, maneuvers, and termination as well as probability density functions for report errors, track initiation and

termination. Define

$$c_{i_1 \cdots i_{N+1}} = -\ln L_{i_1 \cdots i_{N+1}},$$

$$z_{i_1 \cdots i_{N+1}} = \begin{cases} 1 & \textit{if}(z_{i_1} \cdots {}_{i_{N+1}}) \textit{ are assigned to as a track,} \\ 0 & \textit{otherwise.} \end{cases} \qquad [(3.4)]\,(1.7)$$

Amend the paragraph beginning at column 48, line 6, as follows:

Then recognizing that,

$$-\ln\left[\frac{P([\Gamma\gamma]\Gamma = \gamma \mid Z^{N+1})}{P([\gamma^0]\Gamma = \gamma^0 \mid Z^{N+1})}\right] = [\textstyle\sum]\sum_{(i_1,\ldots,i_{N+1})\in\gamma[\ c]} c_{i_1 \ldots i_{N+1}},$$

the problem (3.1) can be expressed as the [N+1-dimensional] $(N+1)$-dimensional assignment problem

$$\textit{Minimize} \quad \sum_{i_1=0}^{M_1} \cdots \sum_{i_{N+1}=0}^{M_{N+1}} c_{i_1 \ldots i_{N+1}} z_{i_1 \ldots i_{N+1}}$$

$$\textit{Subject To} \quad \sum_{i_2=0}^{M_2} \cdots \sum_{i_{N+1}=0}^{M_{N+1}} z_{i_1 \ldots i_{N+1}} = 1, \quad i_1 = 1,\ldots,M_1, \qquad (3.5)$$

$$\sum_{i_1=0}^{M_1} \cdots \sum_{i_{k-1}=0}^{M_{k-1}} \sum_{i_{k+1}=0}^{M_{k+1}} \cdots \sum_{i_{N+1}=0}^{M_{k+1}} z_{i_1 \ldots i_{N+1}} = 1$$

$$\textit{for } i_k = 1,\ldots,M_k \textit{ and } k = 2,\ldots,N,$$

$$\sum_{i_1=0}^{M_1} \cdots \sum_{i_{N+1-1}=0}^{M_N} z_{i_1 \ldots i_{N+1}} = 1, \; i_{N+1} = 1,\ldots,M_{N+1},$$

$$z_{i_1 \ldots i_{N+1}} \in \{0,1\} \textit{ for all } i_1,\ldots,i_{N+1},$$

where $c_{0 \ldots 0}$ is arbitrarily defined to be zero. Here, each group of sums in the constraints represents the fact that each non-dummy report occurs exactly once in a "track of data." One can modify

this formulation to include [multiassignments] multi-assignments of one, some, or all the actual reports. The assignment problem (3.5) is changed accordingly. For example, if $z_{i_k}^k$ is to be assigned no more than, exactly, or no less than $n_{i_k}^k$ times, then the "=1" in the constraint (3.5) is changed to "≤, =, ≥ $n_{i_k}^k$," respectively. In making these changes, one must pay careful attention to the independence assumptions, which need not be valid in many applications. Expressions for the likelihood ratios can be found in the work of [Poore, 1994, *Computational Optimization & Appl.*, 3:27-57,] (A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. Cox, P. Hansen, and B. Julesz, editors. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, R.I., 19:169-198, 1995)* and the references therein.

Amend the paragraph heading beginning at column 48, line 47, as follows:

III.4. Track Initiation and Maintenance

Amend the paragraph heading beginning at column 48, line 58, as follows:

III.4.1 The First Approach to Track Maintenance and Initiation

Amend the paragraph beginning at column 48, line 59, as follows:

The first method as explained in this section is an improved version of our first track maintenance scheme (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994) and uses the same window length for track initiation and maintenance after the initialization step. The process is to start with a window of length *N*+1 anchored at frame one. In the

first step [one] there is only one track initiation in that we assume no prior existing tracks. In the second and all subsequent frames, there is a window of length N anchored at frame k plus a collection of tracks up to frame k. This window is denoted by $\{k[;\ ],\ k+1, \ldots, k+N\}$. The following explanation of the steps is much like mathematical induction in that we explain the first step and then step k to step $k+1$.

Amend the paragraph beginning at column 49, line 4, as follows:

Track Maintenance and Initiation: Step 1. Let

$$\{i_1(l_2), i_2(l_2) \ldots, i_N(l_2), i_{N+1}(l_2)\}_{l_2=1}^{L_2} \qquad ([4.1]]\underline{3.6})$$

be an enumeration of all those zero-one variables in the solution of the assignment problem (3.5) (i.e., $z_{i_1 i_2 \cdots i_{N+1}=1}$) excluding all the false reports in the solution (i.e., all those zero-one variables with exactly one nonzero index) and zero-one variables in the solution for which $(i_1, i_2) = (0,0)$. (The latter can correspond to tracks that initiate on frames three and higher.) These denote our initial tracks.

Amend the paragraph beginning at column 49, line 14, as follows:

Consider only the first two index sets in this enumeration ([4.1]]<u>3.6</u>) and add the zero index $l_2=0$ with the corresponding values of i_1 and i_2 being zero. Thus, the enumeration is now $\{i_1(l_2), i_2(l_2)\}_{l_2=0}^{L_2}$. The notation $T_2(l_2) \equiv (z_{i_1(l_2)}, z_{i_2(l_2)})$ will be used for this pairing. Suppose now that the next data set, i.e., the $(N+2)^{th}$ set, is added to the problem.

Amend the paragraph beginning at column 49, line 20, as follows:

To explain the costs for the new problem, one starts with the hypothesis that a partition $\gamma \in \Gamma*$ being true is now conditioned on the truth of the pairings on the first two frames

being correct. Corresponding to the sequence $\left\{T_2(l_2), z_{i_3}, \ldots, z_{i_{n+2}}\right\}$, the likelihood function is then given by

[4.2a] (3.7)

$$L_\gamma \equiv \prod_{\left\{T_2(l_2), z_{i_3}, \ldots, z_{i_{N+2}}\right\} \in \gamma} L_{l_2 i_3 \ldots i_{N+2}}, \quad \textbf{\textit{where}}$$

$$L_{l_2 i_3 \cdots i_{N+2}} \equiv L_{T_2(l_2)} L_{z_{i_3}, \ldots, z_{i_{N+2}}},$$
$$L_{T_2(0)} = 1.$$

Amend the paragraph beginning at column 49, line 34, as follows:

Next, define the cost and the corresponding zero-one variable

$$C_{l_2 i_3 \cdots i_{N+2}} = -\ln L_{l_2 i_3 \cdots i_{N+2}},$$

$$z_{l_2 i_3 \cdots i_{N+2}} = \begin{cases} 1, \; \textit{if} \left\{z^3_{i_3}, \ldots, z^{N+2}_{i_{N+2}}\right\} \; \textbf{\textit{is assigned to Track}} \; T_2(l_2), \\ 0, \; \textbf{\textit{otherwise,}} \end{cases}$$

([4.2b] 3.8)

respectively. Then the track maintenance problem Maximize $\{L_\gamma | \gamma \in \Gamma^*\}$ can be formulated as the following multidimensional assignment

Minimize $$\sum_{l_2=0}^{L_2} \sum_{i_3=0}^{M_3} \cdots \sum_{i_{N+2}=0}^{M_{N+2}} c_{l_2 i_3 \cdots i_{N+2}} z_{l_2 i_3 \cdots i_{N+2}}$$

Subject to: <u>(3.9)</u>

$$\sum_{i_3=0}^{M_3} \cdots \sum_{i_{N+2}=0}^{M_{N+2}} z_{l_2 i_3 \cdots i_{N+2}} = 1,\ l_2 = 1,\ l_2, \ldots, L_2,$$

$$\sum_{l_2=0}^{L_2} \sum_{i_4=0}^{M_4} \cdots \sum_{i_{N+2}=0}^{M_{N+2}} z_{l_3 i_3 \cdots i_{N+2}} = 1,\ i_3 = 1, \ldots, M_3,$$

$$\sum_{l_2=0}^{L_2} \sum_{i_3=0}^{M_3} \cdots \sum_{i_{p-1}=0}^{M_{p-1}} \sum_{i_{p+1}=0}^{M_{p+1}} \cdots \sum_{i_{N+2}=0}^{M_{N+2}} z_{l_2 i_3 \cdots i_{N+2}} = 1,$$

$$\textit{for } i_p = 1, \ldots, M_p \textit{ and } p = 4, \ldots, N+2-1,$$

$$\sum_{l_2=0}^{L_2} \sum_{i_3=0}^{M_3} z_{i_1 \cdots i_{N+1}} \cdots \sum_{i_{N+2-1}=0}^{M_{N+2-1}} z_{l_2 i_3 \cdots i_{N+2}} = 1,\ i_{N+2} = 1, \cdots, M_{N+2},$$

$$z_{l_2 i_3 \cdots i_{N+2}} \in \{0,1\} \textit{ for all } l_2 i_3, \cdots, i_{N+2}.$$

Amend the paragraph beginning at column 50, line 5, as follows:

Track [Maintence] <u>Maintenance</u> and Initiation: Step k. At the beginning of the k^{th} step, we solve the following $(N+1)$-dimensional assignment problem.

[(4.4)] <u>(3.10)</u>

Minimize[:]

$$\sum_{l_k=0}^{L_k} \sum_{i_{k+1}=0}^{M_{k+1}} \cdots \sum_{i_{k+N}=0}^{M_{k+N}} c_{l_k i_{k+1} \cdots i_{k+N}} z_{l_k i_{k+1}} \cdots i_{k+N}$$

Subject to:

$$\sum_{i_{k+1}=0}^{M_{k+1}} \cdots \sum_{i_{k+N}=0}^{M_{k+N}} z_{l_k i_{k+1} \cdots i_{k+N}} = 1,\ l_k = 1, \ldots L_k,$$

$$\sum_{l_k=0}^{L_k} \sum_{i_{k+2}=0}^{M_{k+2}} \cdots \sum_{i_{k+N}=0}^{M_{k+N}} z_{l_k i_{k+1} \cdots i_{k+N}} = 1,\ i_{k+1} = 1, \ldots M_{k+1},$$

$$\sum_{l_k=0}^{L_k} \sum_{i_{k+1}=0}^{M_{k+1}} \cdots \sum_{i_{p-1}=0}^{M_{p-1}} \sum_{i_{p+1}=0}^{M_{p+1}} \cdots \sum_{i_{k+N}=0}^{M_{k+N}} z_{l_k i_{k+1} \cdots i_{k+N}} = 1,$$

$$\textit{for } i_p = 1, \ldots, M_p \textit{ and } p = k+2, \ldots, N+k-1,$$

$$\sum_{l_k=0}^{L_k} \sum_{i_{k+1}=0}^{M_{k+1}} \cdots \sum_{i_{k+1+N-2}=0}^{M_{k+1+N-2}} z_{l_k i_{k+1} \cdots i_{k+N}} = 1, \; i_{k+N} = 1, \ldots, M_{k+N}$$

$$z_{l_k i_{k+1} \cdots i_{k+N}} \in \{0,1\} \; \textit{for all} \; l_k, i_{k+1}, \ldots, i_{k+N}.$$

where for the first step l_1 and L_1 are replaced by i_1 and M_1, respectively. The first index l_k in the subscripts corresponds to the sequence of tracks $\{T_k(l_k)\}_{l_k=0}^{L_k}$, where $T_k(l_k) = \{z_{i_1}^{1}(l_k), \cdots, z_{i_k}^{k}(l_k)\}$ $T_k(l_k) = \{z_{i_1}^{1}(l_k), \cdots, z_{i_k}^{k}(l_k)\}$ is a track of data from the solution of the problem prior to the formulation of ([4.4]3.10).[or if] If k=1, then it is just the first data set in frame one.

Amend the paragraph beginning at column 50, line 41, as follows:

Basic Assumption. Suppose problem ([4.4]3.10) has been solved and let the solution, i.e., those zero-one variables equal to one, be enumerated by

$$\{(l_k(l_{k+1}), i_{k+1}(l_{k+1}), \ldots, i_{k+N}(l_{k+1}))\}_{l_{k+1}=1}^{L_{k+1}} \qquad ([4.5]3.11)$$

with the following exceptions.

a. All [zero one] zero-one variables for which $(l_k, i_{k+1}) = (0,0)$ are excluded. Thus, tracks that initiate on frames after the $(k+1)^{th}$ are not included in the list.

b. All zero-one variables whose subscripts have the form $l_k=0$ and exactly one nonzero index in the remaining indices $\{i_{k+1}, \ldots, i_{k+N}\}$ are excluded. These correspond to false reports on frames [p=k+1]$p = k+1$, …, [k+N]$k+N$.

c. All variables [$(z_{l_k i_{k+1} \cdots i_{k+N}})$] $z_{l_k i_{k+1} \cdots i_{k+N}}$ for which $(i_{k+1}, \ldots, i_{k+N}) = (0,0,\ldots,0)$ and $i_k(l_k)=0$ [in]are excluded from ([4.5]1.11). In other words, the reports on the last N+1 frames in the $\{T_k(l_k), z_{i_{k+1}}^{k+N}, \ldots, z_{i_{k+N}}^{k+N}\}$ are all dummy. Any solution with this feature corresponds to a terminated track.

Amend the paragraph beginning at column 50, line 58, as follows:

Given the enumeration ([4]3.[5]11), one now fixes the assignments on the first two index sets in the list ([4]3.[5]11). The zero index $l_{k+1}=0$ is added to the enumeration to specify $\left(l_k(0), i_{k+1}(0)\right)=(0,0)$ that is used to represent false reports and tracks that initiate on frame k+2 or later, so that the enumeration ([4.5]3.11) is now

$$\left\{\left(l_k(l_{k+1}), i_{k+1}(l_{k+1})\right)\right\}_{l_{k+1}=0}^{L_{k+1}}. \qquad [(4.6)]$$

Amend the paragraph beginning at column 51, line 9, as follows:

The corresponding hypothesis about a partition $\gamma \in \Gamma^*$ being true is now conditioned on the truth of the L_{k+1} tracks existing at the beginning of the N-frame window. (Thus the assignments prior to this sliding window are fixed.) The likelihood function is given by

$$\begin{aligned} L_{l_{k+1} i_{k+2} \cdots i_{k+1+N}} &= L_{T_{k+1}(l_{k+1})} L_{z_{i_{k+2}}^{k+2}, \ldots, z_{i_{k+1+N}}^{k+1+N}}, \\ L_{T_{k+1}(0)} &= 1. \end{aligned} \qquad ([4.7a]3.12)$$

Amend the paragraph beginning on column 51, line 23, as follows:

Next, define the cost and the zero-one variable by [said at least one inert gas has an amount, on a molar basis, that is greater than the amount of said oxygen in said pressurized medium; (4.7b)]

$$c_{l_{k+1} i_{k+2} \cdots i_{k+1+N}} = -\ln L_{l_{k+1} i_{k+2} \cdots i_{k+1+N}} \equiv -\ln L_{T_{k+1}(l_{k+1}) z_{i_{k+2}} \cdots z_{i_{k+1+N}}},$$

$$z_{l_{k+1} i_{k+2} \cdots i_{k+1+N}} = \begin{cases} 1, & \text{if } \left\{z_{i_{k+2}}^{k+2}, \ldots, z_{i_{k+1+N}}^{k+1+N}\right\} \text{ is assigned to } T_{k+1}(l_{k+1}), \\ 0, & \text{otherwise,} \end{cases} \qquad (3.13)$$

respectively, so that the track extension problem, which was originally formulated as ***Maximize*** $\{L_\gamma | \gamma \in \Gamma^*\}$, can be expressed as exactly the same multi-dimensional assignment in (3.4) [but]with k replaced by $k+1$. Thus, we do not repeat it here.

Amend the paragraph beginning at column 51, line 42, as follows:

III.4.2. The Second Approach to Track Maintenance and Initiation

Amend the paragraph beginning at column 51, line 50, as follows:

The General Step k. To formulate a problem for track initiation and maintenance we consider a moving window centered as frame k of length $I+J+1$ denoted by $[k-I, \ldots, k, \ldots, k+J]$. In this representation, the window length for track maintenance is J and that for initiation, $I+J+1$. The objective will be to explain the situation at this center and then the move to the same length window at center $k+1$ denoted by [[k+1-I]$\underline{k+1-I}$, …, $k+1$, …, [k+1+J]$\underline{k+1+J}$], i.e., by moving the [frame]window to the right one frame at time[to the right]. The explanation from the first step follows hereinafter.

Amend the paragraph beginning at column 51, line 60, as follows:

The notation for a track of data is

$$T_k(l_k) = \left\{ z^1_{i_1(l_k)}, \cdots, z^p_{i_p(l_k)}, \cdots, z^k_{i_k(i_k)} \right\}, \qquad ([4.8]\underline{3.14})$$

where the index l_k is used for an enumeration of those reports paired together. We also use the notation $T_{p,k}(l_k)$ to denote the sequence of reports belonging to track $T_k(l_k)$ but

restricted to frames prior to and including p. Thus,

$$T_k(l_k) \equiv T_{k,k}(l_k),$$
$$T_{p,k}(l_k) = \left\{ z^1_{i_1(l_k)}, \cdots, z^p_{i_p(l_k)} \right\},$$
$$T_{k,k}(l_k) = T_{p,k}(l_k) \cup \left\{ z^{p+1}_{i_{p+1}(l_k)}, \cdots, z^{k-1}_{i_{k-1}(l_k)}, z^k_{i_k(l_k)} \right\} \text{ for any } p \le k-1.$$

([4.9]<u>3.15</u>)

Amend the paragraph beginning at column 52, line 7, as follows:

Given this notation for the tracks and partition of the data in the frames $\{k-I, \ldots, k, \ldots, k+J\}$, $L_{T_{p,k}(l_k)}$ will denote the accumulated likelihood ratio up to and including frame $p(p \le k)$ for a track that is declared as existing on frame k as a solution of the assignment problem. In this notation, the likelihood for $T_{p,k}(l_k)$ and that of the association of $\left\{ z^{k+1}_{i_{k+1}}, \cdots, z^{k+N}_{i_{k+N}} \right\}$ with track $T_k(l_k)$ is given by

$$L_{T_k(l_k)} = L_{T_{p,k}(l_k)} L_{i_{p+1}(l_k) \cdots i_k(l_k)} \text{ for any } p \le k-1,$$
$$L_{T_k(l_k) i_{k+1} \cdots i_{k+N}} = L_{T_{p,k}(l_k)} L_{i_{p+1}(l_k) \cdots i_k(l_k) i_{k+1} \cdots i_{k+N}} \text{ for any } p \le k-1,$$

([4.10]<u>3.16</u>)

respectively.

Amend the paragraph beginning at column 52, line 20, as follows:

The cost for the assignment of $\left\{ z^{k+1}_{i_{k+1}}, \cdots, z^{k+N}_{i_{k+N}} \right\}$ to track $T_k(l_k)$ and the corresponding [zero one]<u>zero-one</u> variable are given by

$$c^{T}_{l_k i_{k+1} \cdots i_{k+J}} = -\ln[L_{T_k(l_k)} L_{i_{k+1} \cdots i_{k+J}}]$$
$$= -\ln[L_{T_{p,k}(l_k)} L_{i_{p+1}(l_k) \cdots i_k(l_k) i_{k+1} \cdots i_{k+J}}]$$ and
$$z^{T}_{l_k i_{k+1} \cdots i_{k+J}} = \begin{cases} 1, & \textit{if } \left\{ z^{k+1}_{i_{k+1}}, \ldots, z^{k+N}_{i_{k+N}} \right\} \textit{ is assigned to track } T_k(l_k), \\ 0, & \textit{otherwise}, \end{cases} \qquad ([4.11]\underline{3.17})$$

respectively. Likewise, for costs associated with the false reports on the frames $k-I$ to k and as associated with $\left\{ z^{k+1}_{i_{k+1}}, \ldots, z^{k+J}_{i_{k+J}} \right\}$ and the corresponding zero-one [variable]<u>variables</u> are given by:

$$c^{F}_{i_{k-1} \cdots i_k i_{k+1} \cdots i_{k+J}} = -\ln[L_{i_{k-1} \cdots i_k i_{k+1} \cdots i_{j+J}}] \qquad ([4.12]\underline{3.18})$$

$$z^{F}_{i_{k-1} \cdots i_k i_{k+1} \cdots i_{k+J}} = \begin{cases} 1, & \textit{if } \{ z_{i_{k-I}}, \ldots, z_{i_k i_{k+1}}, \ldots, z_{i_{k+J}} \} \\ & \quad \textit{are assigned as a track}, \\ 0, & \textit{otherwise}. \end{cases}$$

For the window of frames in the range $[[k-I,\ldots,k]]\underline{\{k-I,\ldots,k\}}$, the easiest explanation of the partition of the reports is based on the definitions

[(4.13)]

$$T_{p,k} = \left\{ \begin{array}{l} i_p \mid z^{p}_{i_p} \textit{ belongs to one of the tracks} \\ \quad \textit{listed on frame } k\textit{, i.e., to one} \\ \quad \textit{of the } T_k(l_k) \textit{ for some } l_k = 1, \ldots, l_k \end{array} \right\}, \qquad \underline{(3.19)}$$

$$F_{p,k} = (\{1, \ldots, M_p\} \backslash T_{p,k}) \cup \{0\},$$

so that all of those reports not used in the tracks $T_k(l_k)$ on frame p are put into the set of available reports $F_{p,k}$ for the formation of tracks over the entire window. Thus, we formulate the assignment problem as

$$\begin{aligned} \text{Minimize} \quad & \sum_{p=k-I}^{k} \sum_{i_p \in F_{p,k}} \sum_{r=k+1}^{k+J} \sum_{i_r=0}^{M_r} c^F_{i_{k-I} \cdots i_k i_{k+1} \cdots i_{k+J}} z^F_{i_{k-I} \cdots i_k i_{k+1} \cdots i_{k+J}} \\ & + \sum_{l_k=1}^{L_k} \sum_{r=k+1}^{k+J} \sum_{i_r=0}^{M_r} c^T_{l_k i_{k+1} \cdots i_{k+J}} z^T_{l_k i_{k+1} \cdots i_{k+J}} \\ \text{Subject To} \quad & \sum_{\{p=k-I, p \neq q\}}^{k} \sum_{i_p \in F_{p,k}} \sum_{r=k+1}^{k+J} \sum_{i_r=0}^{M_r} z^F_{i_{k-I} \cdots i_k i_{k+1} \cdots i_{k+J}} = 1 \\ & \text{for } i_q \in F_{p,k} \backslash \{0\} \text{ and } q = k-I, \ldots, k, \\ & \sum_{p=k-I}^{k} \sum_{i_p \in F_{p,k}} \sum_{\{r=k+1, r \neq q\}}^{k+J} \sum_{i_q=0}^{M_r} z^F_{i_{k-I} \cdots i_k i_{k+1} \cdots i_{k+J}} = 1 \\ & \text{for } i_q = 1, \ldots, M_q \text{ and } q = k+1, \ldots, k+J, \\ & z_{i_{k-I} \cdots i_k i_{k+1} \cdots i_{k+J}} \in \{0,1\} \text{ for all } i_{k-I}, \ldots, i_k, i_{k+1}, \ldots, i_{k+J}, \\ & \sum_{r=k+1}^{k+J} \sum_{i_r=0}^{M_r} z^T_{l_k i_{k+1} \cdots i_{k+J}} = 1, \quad l_k = 1, \ldots, L_k, \\ & \sum_{l_k=1}^{L_k} \sum_{\{r=k+1, r \neq q\}}^{k+J} \sum_{i_r=0}^{M_r} z^T_{l_k i_{k+1} \cdots i_{k+J}} = 1, \quad i_q = 1, \ldots, M_q \\ & \text{for } q = k+1, \ldots, k+J, \\ & z_{l_k i_{k+1} \cdots i_{k+J}} \in \{0,1\} \text{ for all } l_k \geq 1, i_{k+1}, \ldots, i_{k+J}. \end{aligned} \tag{3.20}$$

Amend the paragraph beginning at column 53, line 21, as follows:

Basic Assumptions. Suppose problem ([4.14]1.20) has been solved and let the solution, i.e.[,] those zero-one variables equal to one, be enumerated by

$$\begin{aligned} & \left\{ z^T_{l_k(l_{k+1}) i_{k+1}(l_{k+1}) \cdots i_{k+J}(l_{k+1})} \right\}_{l_{k+1}=1}^{\tilde{L}_{k+1}}, \\ & \left\{ z^F_{i_{k-I}(l_{k+1}) \cdots i_k(l_{k+1}) i_{k+1}(l_{k+1}) \cdots i_{k+J}(l_{k+1})} \right\}_{l_{k+1}=\tilde{L}_{k+1}+1}^{L_{k+1}} \end{aligned} \qquad ([4.15]\underline{3.21})$$

with the following exceptions.

a. All [zero one] zero-one variables in the second list for which $(i_{k-I}, \ldots, i_k, i_{k+1}) = (0, \ldots, 0, 0)$ are excluded. Thus, tracks that initiate on frames after the $(k+1)^{th}$ are not included in the list.

b. All false reports [axe] <u>are</u> excluded, i.e., all zero-one variables in the second list whose subscripts have exactly one nonzero index.

c. All variables $z^{T}_{l_k i_{k+1} \cdots i_{k+N}}$ for which $(i_{k+1}, \cdots, i_{k+N}) = (0, 0, \ldots, 0)$ and $Z(p) \cap T_k(l_k) = 0$ for $p=k-K, \ldots, k$ where [$K \geq 0$]<u>$K \geq 0$</u> is user specified. Thus the track $T_k(l_k)$ is terminated if it is not observed over $K+J+1$ frames.

Given the enumeration ([4.15]<u>3.20</u>), one now fixes the assignments on the all index sets up to and including the $(k+1)^{th}$ index sets.

[(4.16)]

$$T_{k+1}(l_{k+1}) = \begin{cases} \left(T_k(l_k(l_{k+1})), z^{k+1}_{i_{k+1}}(l_{k+1}) \right) \\ \quad for\ l_{k+1} = 1, \ldots, \tilde{L}_{k_{1+1}}, \\ \left(z^{k-I}_{i_{k-I}(l_{k+1})}, \ldots, z^{k}_{l_k(l_{k+1})}, z^{k+1}_{i_{k+1}(l_{k+1})} \right) \\ \quad for\ l_{k+1} = \tilde{L}_{k+1} + 1, \ldots, L_{k+1}. \end{cases} \qquad \underline{(3.22)}$$

Amend the paragraph beginning at column 53, line 52, as follows:

Thus one can now formulate the assignment problem for the next problem exactly as in ([4-14]<u>3.20</u>) but with k replaced by $k+1$. Thus, we do not repeat it here.

Amend the paragraph beginning at column 54, line 7, as follows:

If [$I+J>N$]<u>$I+J > N$</u>, then one possibility is to start the process with $N+1$ frames, and assuming $J \leq N$, proceed as before replacing I by $N-J$ for the moment, and continue to add frames without lopping off the first frame in the window until

reaches a window of length $I+J+1$. Then we proceed as in the previous paragraph.

Amend the paragraph beginning at column 54, line 14, as follows:

If [$I+J<N$] $\underline{I+J \le N}$, then one can solve the track initiation problem (3.5), formulate the problem with the center of the window at [$k+1=N+1-J$] $\underline{k+1 = N+1-J}$, enumerate the solutions as above, and lop off the first $N-J-I$ frames. Then, we proceed just as in the case $I+J=N$.

Insert the following heading before the paragraph beginning at column 54, line 37:

III.5. CONCLUDING COMMENTS

Amend the paragraph beginning at column 54, line 37 as follows:

Suppose we have solved problem ([4.4]3.10) and have enumerated all those zero-one variables in the solution of ([4.4]3.10) as in ([4.5]3.11). Add the zero index $l_{k+1}=0$, so that the enumeration is

$$\left\{\left(l_k(l_{k+1}), i_{k+1}(l_{k+1}), \ldots, i_{k+N}(l_{k+1})\right)\right\}_{l_{k+1}=0}^{L_{k+1}}. \qquad ([5.1]\underline{3.23})$$

With this enumeration one can define the cost by

$$c_{l_{k+1} i_{k+1+N}} = c_{l_k(l_{k+1})[,] i_{k+1}(l_{k+1})[,\ldots,i_]\ldots i_{N+1}(l_{k+1}) i_{k+1+N}} \qquad ([5.2]\underline{3.24})$$

and the [two dimensional] two-dimensional assignment problem

[(5.3)] (3.25)

$$\Phi_2 \equiv Minimize \sum_{l_{k+1}=0}^{L_{k+1}} \sum_{i_{k+1+N}=0}^{M_{k+1+N}} c^2_{l_{k+1}i_{k+1+N}} z^2_{l_{k+1}i_{k+1+N}} \equiv V_2(z^2)$$

$$Subject\ to \sum_{i_{k+1+N}=0}^{M_k+1+N} z^2_{l_{k+1}i_{k+1+N}} = 1,\ l_{k+1} = 1,\ldots,L_{k+1},$$

$$\sum_{l_{k+1}=0}^{L_{k+1}} z^2_{l_{k+1}i_{k+1+N}} = 1,\ i_{k+1+N} = 1,\ldots,M_{k+1+N},$$

$$z^2_{l_{k+1}i_{k+1+N}} \in \{0,1\}\ for\ all\ l_{k+1}, i_{k+1+N}.$$

Let w be an optimal or feasible solution to this two-dimensional assignment problem and define

([5.4]3.26)

$$\hat{z}_{i_{k+1}\cdots i_{k+N}i_{k+1+N}} = \begin{cases} 1, & if\ (i_{k+1},\ldots,i_{k+N}) = \left(i_{k+1}(l_{k+1}),\ldots,i_{k+N}(l_{k+1})\right) \\ & and\ w_{l_{k+1}i_{k+1+N}} = 1\ for\ some\ l_{k+1}=1,\ldots,L_{k+1} \\ & or\ if\ (l_{k+1}, i_{k+1+N}) = (0,0) \\ 0, & otherwise. \end{cases}$$

This need not satisfy the constraints in that there are usually many objects left unassigned. Thus, one can complete the assignment by using the zero-one variables in ([4.4]3.10) with k replaced by k+1 with exactly one nonzero index corresponding to any unassigned object or data report.

Amend the paragraph beginning at column 55, line 16, as follows:

For the dual solutions, the multipliers arising from the solution of the [two dimensional] two-dimensional assignment problem ([5.3]3.25) corresponding to the second variable, i.e., $\left\{u^{k+1+N}_{i_{k+1+N}}\right\}^{M_{k+1+N}}_{i_{k+1+N}=1}$. These are good initial values [for]to use in a relaxation scheme [[11,12]](A.B. Poore and N.Rijavec. Partitioning multiple data sets: multidimensional assignments and Lagrangian relaxation, in quadratic assignment and related problems; P.M. Pardalos and H.

Wolkowicz, editors, *DIMACS series in Discrete Mathematics and Theoretical Computer Science*, 16:25-37, 1994; A.J. Robertson III. A class of lagrangian relaxation algorithms for the multidimensional assignment problem. *Ph.D. Thesis, Colorado State University, Ft. Collins, CO*, 1995). Finally, note that one can also develop a warm start for problem ([4.14]3.20) in a similar fashion.

Insert the following heading before the paragraph beginning at column 55, line 22:

IV. REVIEW OF NEW RELAXATION SCHEMES

IV.1. Introduction

Multidimensional assignment problems govern the central problem of data association in multisensor and multitarget tracking, i.e., the problem of partitioning observations from multiple scans of the surveillance region and from either single or multiple sensors into tracks and false alarms. This fundamental problem can be stated as (A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. Cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, R.I.*, 19:169-198, 1995; A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994)

$$\text{Minimize} \quad \sum_{i_1=0}^{M_1} \cdots \sum_{i_N=0}^{M_N} c_{i_1 \cdots i_N} z_{i_1 \cdots i_N} \qquad [(1.1)](4.1)$$

Subject to:

$$\sum_{i_2=0}^{M_2} \cdots \sum_{i_N=0}^{M_N} z_{i_1 \cdots i_N} = 1, i_1 = 1, \ldots, M_1,$$

$$\sum_{i_1=0}^{M_1} \cdots \sum_{i_{p-1}=0}^{M_{p-1}} \sum_{i_{p+1}=0}^{M_{p+1}=0} \cdots \sum_{i_N=0}^{M_N} z_{i_1 \cdots i_N} = 1$$

$$\text{for } i_p = 1, \ldots, M_p \text{ and } p = 2, \ldots, N-1,$$

$$\sum_{i_1=0}^{M_1} \cdots \sum_{i_{n-1}=0}^{M_{N-1}} z_{i_1 \cdots i_N} = 1, \; i_N = 1, \ldots, M_N,$$

$$z_{i_1 \cdots i_N} \in \{0,1\} \text{ for all } i_1, \ldots, i_N,$$

[for $i_k=1,\ldots,M_k$ and $k=2,\ldots,N$,

$zi_1 \cdots 1_{N+1} \in \{0,1\}$ for all $i_1, \cdots, i_{N+1}$]

where $c_{0 \cdots 0}$ is arbitrarily defined to be zero and is included for notational convenience. One can modify this formulation to include [multiassignments]*multi-assignments* of one, some, or all of the actual reports. The zero index is used in representing missing data, false alarms, initiating and terminating tracks as described in (A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. Cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, R.I.,* 19:169-198, 1995; A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications,* 3:27-57, 1994). In these problems, we assume that all zero-one variables $z_{i_1 \cdots i_N}$ with precisely one nonzero index are free to be assigned and that the corresponding cost coefficients are well-defined. (This is a valid assumption in the tracking environment (A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets,

I.J. Cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, R.I.,* 19:169-198, 1995; A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications,* 3:27-57, 1994).) Although not required, these cost coefficients with exactly one nonzero index can be translated to zero by cost shifting (A.B. Poore and N. Rijavec. A lagrangian relaxation algorithm for multidimensional assignment problems arising form multitarget tracking. *SIAM Journal of Optimization,* 3, No. 3:544-563, 1993) without changing the optimal assignment. Finally, this formulation is of sufficient generality to include the symmetric problem and the asymmetric inequality problem[.] (A.J. Robertson III. A class of lagrangian algorithms for the multidimensional assignment problem. *Ph.D. Thesis, Colorado State University, Ft. Collins, CO,* 1995).

Amend the paragraph beginning at column 55, line 64, as follows:

The data association problems in tracking that are formulated as Equation ([1.1]4.1) have several characteristics. They are normally sparse, the cost coefficients are noisy and the problem is NP-hard (M.R. Galey and D.S. Johnson. *Computers and Intractability.* W.H. Freeman and Company, San Francisco, CA, 1979), but it must be solved in real-time. The only known methods for solving this NP-hard problem optimally are enumerative in nature, with branch-and-bound being the most efficient; however, such methods are much too slow for real-time applications. Thus

one must resort to suboptimal approaches. Ideally, any such algorithm should solve the problem to within the noise level, assuming, of course, that one can measure this noise level in the physical problem and that the mathematical method provides a way to decide if the criterion has been met.

Amend the paragraph beginning at column 56, line 10, as follows:

There are many algorithms that can be used to construct suboptimal solutions to NP-hard combinatorial optimization problems. These include greedy (and its many variants), relaxation, simulated annealing, tabu search, genetic algorithms, and neural [netork] algorithms (C.H. Papadimitriou and K.Steiglitz. *Combinatorial Optimization: Algorithm and Complexity.* Prentice-Hall, Inc., Englewood Cliffs, NJ, 1982; J.Pearl. *Heuristics: Intelligent Search Strategies for Computer Problem Solving.* Addison-Wesley, Reading, MA, 1984; C.R. Reeves ed. *Modern Heuristic Techniques for Combinatorial Problems.* Halstead Press, Wiley, NY, 1993). For the [three dimensional] three- dimensional assignment problem, Pierskalla (W. Pierskalla. The tri-substitution method for three-dimensional assignment problem. *Journal du CORS,* 5:71-81, 1967) developed the tri-substitution method, which is a variant of the simplex method. Frieze and Yadegar (A.M. Frieze and J. Yadegar. An algorithm for solving 3-dimensional assignment problems with application to scheduling a teaching practice. *Journal of the Operational Research Society,* 39:989-955, 1981) introduced a method based on Lagrangian relaxation in which a feasible solution is *recovered* using information provided by the relaxed solution. Our choice of approaches is strongly

influenced by the need to balance real-time performance and solution quality. Lagrangian relaxation based methods have been used successfully in prior tracking applications[.] (K.R. Pattipati, S. Deb, and Y. Bar-Shalom. A s-dimensional assignment algorithm for track initiation. In *Proceedings of the IEEE Systems Conference, Kobe, Japan,* pages 127-130, 1992; Y. Bar-Shalom, S. Deb, K.R. Pattipati, and H. Tsanakis. A new algorithm for the generalized multidimensional assignment problem. In *Proceedings of the IEEE International Conference on Systems, Math, and Cybernetics, Chicago,* pages 132-136, 1992; A.B. Poore and N. Rijavec. A numerical study of some data association problems arising in multitarget tracking. *Large Scale Optimization: State of the Art, W.W. Hager, D.W. Hearn and P.M. Pardalos, editors. Kluwer Academic Publishers B.V., Boston,* pages 339-361, 1994; A.B. Poore and N.Rijavec. Partitioning multiple data sets: multidimensional assignments and lagrangian relaxation. In P.M. Pardalos and H. Wolkowicz, editors, *Quadratic assignment and related problems: DIMACS Series in Discrete Mathematics and Theoretical Computer Science,* volume 16, pages 25-37, 1994; A.B. Poore and N. Rijavec. A lagrangian relaxation algorithm for multidimensional assignment problems from multitarget tracking. *SIAM Journal of Optimization,*3, No. 3:544-563, 1993). An advantage of these methods is that they provide both an upper and lower bound on the optimal solution, which can then be used to measure the solution quality. These works extend the method of Frieze and Yadegar (A.M. Frieze and J.Yadegar. An algorithm for solving 3-dimensional assignment problems with application to scheduling a teaching practice. *Journal of the Operational Research Society,* 32:989-995, 1981) to the multidimensional case.

Amend the paragraph beginning at column 56, line 28, as follows:

IV.2. Probabilistic Framework for Data Association.[(ABP)]

The goal of this section is to explain the formulation of the data association problems that governs large classes of data association problems in centralized or hybrid centralized-sensor level multisensor/multitarget tracking. The presentation is brief; technical details are presented for both track initiation and maintenance in the work of [Poore] (A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. Cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, R.I.*, 19:169-198, 1995; A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994). The formulation is of sufficient generality to cover the MHT work of Reid, Blackman and Stein (S.S. Blackman. *Multiple Target Tracking with Radar Applications*. Artech House, Norwood, MA, 1986) and modifications by Kurien (T.Kurien. Issues in the designing of practical multitarget tracking algorithms. In *Multitarget-Multisensor Tracking: Advanced Applications by Y. Bar-Shalom*. Artech House, MA, 1990) to include maneuvering targets. As suggested by Blackman (S.S. Blackman. *Multiple Target Tracking with Radar Applications*. Artech House, Norwood, MA, 1986), this formulation can also be modified to include target features (e.g., size and type) into the scoring function. The recent work [of Poore and Drummond] (A.B. Poore

and O.E. Drummond. Track initiation and maintenance using multidimensional assignment problems. In D.W. Hearn, W.W. Hager, and P.M. Pardalos, editors, *Network Optimization*, volume 450, pages 407-422, Boston, 1996. Kluwer Academic Publishers B.V.) significantly extends the work of this section to new approaches for multiple sensor centralized tracking. Future work will involve extensions to track-to-track correlation.

Amend the paragraph beginning at column 56, line 44, as follows:

The data association problems for multisensor and multitarget tracking considered in this work are generally posed as that of maximizing the posterior probability of the surveillance region (given the data) according to

$$\textit{Maximize} \left\{ P(\Gamma=\gamma \mid Z^N) \mid \gamma \in \Gamma^* \right\}, \qquad ([2.1]4.2)$$

where Z^N represents N data sets, γ is a partition of indices of the data (and thus induces a partition of the data), [Γδ

Γ

Γδ γ^0]

[$P(\Gamma=\gamma|Z^N)$] Γ^* is the finite collection of all such partitions, Γ is a discrete random element defined on Γ^*, and $P(\Gamma = \gamma|Z^N)$ is the posterior probability of a partition γ being true given the data Z^N. The term partition is defined below; however, this framework is currently sufficiently general to cover set packings and coverings (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994; A.B. Poore. Multidimensional assignments and

multitarget tracking, partitioning data sets, I.J. cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, R.I.*, 19:169-198, 1995).

Amend the paragraph beginning at column 56, line 64, as follows:

Consider N data sets $Z(k)$ (k=1,…,N) each of M_k reports $\left\{z_{i_k}^{k}\right\}_{i_k=1}^{M_k}$, and let Z^N denote the cumulative data set defined by

$$Z(k) = \left\{z_{i_k}^{k}\right\}_{i_k=1}^{M_k} \quad and \; Z^N = \left\{Z(1),\ldots,Z(N)\right\}, \qquad ([2.2]\underline{4.3})$$

respectively. In multisensor data fusion and multitarget tracking the data sets $Z(k)$ may represent different classes of objects, and each data set can arise from different sensors. For track initiation the objects are measurements that must be partitioned into tracks and false alarms. In our formulation of track maintenance (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994; A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, R.I.*, 19:169-198, 1995), which uses a moving window over time, one data set will be tracks and remaining data sets will be measurements which are

assigned to existing tracks, as false measurements, or to initiating tracks. In sensor level tracking, the objects to be fused are tracks (S.S. Blackman. *Multiple Target Tracking with Radar Applications.* Artech House, Norwood, MA, 1986). In centralized fusion (S.S. Blackman. *Multiple Target Tracking with Radar Applications.* Artech House, Norwood, MA, 1986), the objects may all be measurements that represent targets or false reports, and the problem is to determine which measurements emanate from a common source.

Amend the paragraph beginning at column 57, line 14, as follows:

We specialize the problem to the case of set partitioning (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications,* 3:27-57, 1994; A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. Cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, R.I.,* 19:169-198, 1995) defined in the following way. First, for notational convenience in representing tracks, we add a *zero index* to each of the index sets in Equation ([2.2]4.3), a *dummy report* z_0^k to each of the data sets $Z(k)$ in Equation ([2]4.[2]3), and define a "track of data" as $\left(z_{i_1}^1, \ldots, z_{i_N}^N \right)$, where i_k can now assume the [values] value of 0 [and respectively]. A partition of the data will refer to a collection of tracks of data wherein each report occurs exactly once in one of the tracks of data and such that all data is used up; the occurrence of dummy report is

unrestricted. The dummy report z_0^k serves several purposes in the representation of missing data, false reports, initiating tracks, and terminating tracks (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994; A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, R.I.*, 19:169-198, 1995). The reference partition is that in which all reports are declared to be false.

Amend the paragraph beginning at column 57, line 28, as follows:

Next under appropriate independence assumptions one can show (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994; A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, R.I.*, 19:169-198, 1995)

$$\frac{P(\Gamma=\gamma|Z^N)}{P(\Gamma=\gamma^0|Z^N)} \equiv L_\gamma \equiv \prod_{(i_1 \cdots i_N)\in\gamma} L_{i_1 \ldots i_N}, \qquad ([2.3]\underline{4.4})$$

where γ^0 is a reference partition, $L_{i_1 \ldots i_N}$ is a likelihood ratio containing probabilities for detection, maneuvers, and

termination as well as probability density functions for measurement errors, track initiation and termination. Then if $c_{i_1 \cdots i_N} = -\ln L_{i_1 \cdots i_N}$,

$$-\ln\left[\frac{P(\Gamma=\gamma|Z^N)}{P(\Gamma=\gamma^0|Z^N)}\right] = \sum_{(i_1,\ldots,i_N)\in\gamma} c_{i_1 \ldots i_N}. \qquad ([2.4]\underline{4.5})$$

Using Equation ([2.3]4.4) and the zero-one variable $z_{i_1 \cdots i_N} = 1$ *if* $(i_1, \cdots, i_N) \in \gamma$ and 0 otherwise, one can then write the problem ([2.1]4.4) as the following N-dimensional assignment problem:

[(2.5)]

$$\begin{aligned}
&\textit{Minimize} && \sum_{i_1 \ldots i_N} c_{i_1 \ldots i_N} z_{i_1 \ldots i_N} \\
&\textit{Subject To} && \sum_{i_2 i_3 \cdots i_N} z_{i_1 \ldots i_N} = 1 \quad (i_1 = 1, \ldots, M_1), \qquad \underline{(4.6)} \\
&&& \sum_{i_1 i_3 \cdots i_N} z_{i_1 \ldots i_N} = 1 \quad (i_2 = 1, \ldots, M_2), \\
&&& \sum_{i_1 \ldots i_{p-1} i_{p+1} \ldots i_N} z_{i_1 \ldots i_N} = 1 \quad (i_p = 1, \ldots, M_p \textit{ and } p = 2, \ldots, N-1), \\
&&& \sum_{i_1 i_2 \ldots i_{N-1}} z_{i_1 \ldots i_N} = 1 \quad (i_N = 1, \ldots, M_N), \\
&&& z_{i_1 \ldots i_N} \in \{0,1\} \quad \textit{for all} \quad i_1, \ldots, i_N,
\end{aligned}$$

where $c_{0 \cdots 0}$ is arbitrarily defined to be zero. Here, each group of sums in the constraints represents the fact that each-non-dummy report occurs exactly once in a "track of data." One can modify this formulation to include [multiassignments]*multi-assignments* of one, some, or all the

actual reports. The assignment problem Equation ([2.5]4.6) is changed accordingly. For example, if $z_{i_k}^k$ is to be assigned no more than, exactly, or no less than $n_{i_k}^k$ times, then the "=1" in the constraint ([2.5]4.6) is changed to ≤, =, ≥$n_{i_k}^k$," respectively. Modifications for group tracking and multi-resolution features of the surveillance region will be addressed in future work. In making these changes, one must pay careful attention to the independence assumptions that need not be valid in many applications.

Amend the paragraph beginning at column 58, line 12, as follows:

Expressions for the likelihood ratios $L_{i_1,\dots,i_N}[,]$ can be found in the work of Poore (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994; A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, R.I.*, 19:169-198, 1995). These expressions include the developments of Reid, Kurien (T. Kurien. Issues in the designing of practical multitarget tracking algorithms. In *Multitarget-Multisensor Tracking: Advanced Applications by Y. Bar-Shalom*. Artech House, MA, 1990), and Stein and Blackman (S.S. Blackman. *Multiple Target Tracking with Radar Applications*. Artech House, Norwood, MA, 1986). What's more, they are easily modified to include target features and to account for different sensor types. In *track initiation*, the *N* data sets all represent reports from *N*

sensors, possibly all the same. For *track maintenance*, we use a *sliding window* of N data sets and one data set containing established tracks. (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994; A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, R.I.*, 19:169-198, 1995). The formulation is the same as above except that the dimension of the assignment problem is now N+1.[@@@]

Delete the paragraphs beginning at column 58, line 23, thru column 67, line 35:

Amend the paragraph beginning at column 67, line 36, as follows:

Overview of the Lagrangian Relaxation Algorithm. [(ABP)]

Amend the paragraph beginning at column 67, line 36, as follows:

Having discussed the N-dimensional assignment problem [(1.1)] (3.1), we now turn to a description of the Lagrangian relaxation algorithm. The algorithm will proceed iteratively for a loop k=1,. . ., N-2. At the completion, there remains one two-dimensional assignment problem that provides the last step which yields an optimal (sometimes) or near-optimal solution to the original N-dimensional assignment problem. In step k of this loop (summarized in [Section 4]Section IV.3) one starts with the following [($N-k+1$)]($N-k+1$)-dimensional assignment problem with one change in notation.

If k=1, then the index notation l_1 and L_1 are to be replaced by i_1 and M_1, respectively.

[(3.1)]

$$\text{Minimize} \quad \sum_{l_k i_{k+1} \ldots i_N} c^{N-k+1}_{l_k i_{k+1} \ldots i_N} z^{N-k+1}_{l_k i_{k+1} \ldots i_N}$$

$$\text{Subject To} \quad \sum_{i_{k+1} \ldots i_N} z^{N-k+1}_{l_k i_{k+1} \ldots i_N} = 1, \; l_k = 1, \ldots, L_k, \qquad \underline{(4.7)}$$

$$\sum_{l_k i_{k+2} \ldots i_N} z^{N-k+1}_{l_k i_{k+1} i_{k+2} \ldots i_N} = 1, \; i_{k+1} = 1, \ldots, M_{k+1},$$

$$\sum_{l_k i_{k+1} \ldots i_{p-1} i_{p+1} \ldots i_N} z^{N-k+1}_{l_k i_{k+1} \ldots i_N} = 1,$$

$$\text{for } i_p = 1, \ldots, M_p \text{ and } p = k+2, \ldots, N-1,$$

$$\sum_{l_k i_{k+1} \ldots i_{N-k+1}} z^{N-k+1}_{l_k i_{k+1} \ldots i_N} = 1, \; i_N = 1, \ldots, M_N,$$

$$z_{l_k i_{k+1} \ldots i_N} \in \{0,1\} \text{ for all } l_k, i_{k+1}, \ldots, i_N.$$

To ensure that a feasible solution of Equation *([3.1]]*4.7*) always exists for a sparse problem, all variables with exactly one nonzero index (i.e., variables of the form* $z^{N-k+1}_{l_k 0 \cdots 0}$ *for* [$l_k, \ldots, L_k$] *for* $l_k = 1, \ldots, L_k$ *and* $z^{N-k+1}_{0 \cdots i_p 0 \cdots 0}$ *for* $i_p = 1, \ldots, M_{i_p}$ *and* $p = k+1, \ldots, N$*) are assumed free to be assigned with the corresponding cost coefficients being well-defined.* This assumption is valid in the tracking environment (A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, R.I.*, 19:169-198, 1995; A.B. Poore. Multidimensional assignment formulation of data association

problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994).

Amend the paragraph beginning at column 68, line 26, as follows:

[Subsection]Section IV.3.1 presents some of the properties associated with the Lagrangian relaxation of ([3.1]4.7) based on relaxing the last $(N-k)$-sets of constraints to a two-dimensional one. [A]Section IV.3.2 describes a new approach to the problem of recovering a high quality feasible solution of the original $(N-k+1)$-dimensional problem given a feasible solution (optimal or suboptimal) of the relaxed two-dimensional problem is described hereinafter. A summary of the relaxation algorithm is given [hereinafter] in Section IV.3.3, [following which] and in Section IV.3.4, we establish the maximization of the Lagrangian dual (an important aspect of the relaxation procedure) to be an unconstrained nonsmooth optimization problem and then present a method for computing the subgradients.

Amend the paragraph beginning at column 68, line 38, as follows:

IV.3.1 T-he Lagrangian Relaxed Assignment Problem[.]

The $(N-k+1)$-dimensional problem ([3.1]4.7) has $(N-k+1)$ sets of constraints. A (M_p+1)-dimensional multiplier vector (i.e., [U^p]$\underline{u^p} \in$ [$\mathbb{R}M_p+1$]$\underline{\mathbb{R}^{M_p+1}}$) associated with the [$p$-th]$\underline{p}^{th}$ constraint set will be denoted by $u^p=(u_0^p, u_1^p, \ldots, u_{M_p}^p)^T$ *with* $u_0^p \equiv 0$ being fixed for each $p = k + 2, \ldots, N$ and included for notational convenience only. Now, the [$(N-k+1)$ dimensional] $\underline{(N-k+1)\text{-dimensional}}$ assignment problem ([3.1]4.7) is relaxed to a two-dimensional assignment

problem by incorporating $(N-k-1)$ sets of constraints into the objective function via the Lagrangian. Although any $(N-k-1)$ constraint sets can be relaxed, we choose the *last* $(N-k-1)$ sets of constraints for convenience. The *relaxed problem* is

[(3.2)]

$$\Phi_{N-k+1}(u^{k+2},\ldots,u^N) \equiv \text{Minimize } \phi_{N-k+1}(z^{N-k+1};u^{k+2},\ldots u^N) \equiv$$

$$\text{Minimize} \sum_{l_k i_{k+1}\cdots i_N} c^{N-k+1}_{l_k i_{k+1}\cdots i_N} z^{N-k+1}_{l_k i_{k+1}\cdots i_N} + \sum_{p=k+2}^{N} \sum_{i_p=0}^{M_p} u^p_{i_p}\left[\sum_{l_k i_{k+1}\cdots i_{p-1} i_{p+1}\cdots i_N} z^{N-k+1}_{l_k i_{k+1}\cdots i_N} -1\right] \equiv \qquad (4.8)$$

$$\text{Minimize} \sum_{l_k i_{k+1}\cdots i_N}\left[c^{N-k+1}_{l_k i_{k+1}\cdots i_N} + \sum_{p=k+2}^{N} \sum_{i_p=0}^{M_p} u^p_{i_p}\right] z^{N-k+1}_{l_k i_{k+1}\cdots i_N}$$

$$- \sum_{p=k-2}^{N} \sum_{i_p=0}^{M_p} u^p_{i_p}$$

$$\text{Subject To} \quad \sum_{i_{k+1} i_{k+2}\cdots i_N} z^{N-k+1}_{l_k i_{k+1} i_{k+2}\cdots i_N} = 1, \quad l_k=1,\ldots L_k,$$

$$\sum_{l_k i_{k+2}\cdots i_N} z^{N-k+1}_{l_k i_{k+1} i_{k+2}\cdots i_N} = 1, \quad i_{k+1} = 1,\ldots,M_{k+1}.$$

One of the major steps in the algorithm is the maximization of $\Phi_{N-k+1}(u^{k+2},\ldots,u^N)$ with respect to the multipliers $(u^{k+2},\ldots,u^N)$. It turns out that Φ_{N-k+1} is a concave, continuous, and piecewise affine function of the multipliers $(u^{k+2},\ldots,u^N)$, so that the maximization of Φ_{N-k+1} is a problem of nonsmooth optimization. Since many of these algorithms[[???]] require a function value and a subgradient of Φ_{N-k+1} at any required multiplier value $(u^{k+2},\ldots,u^N)$, we address this problem in the next subsection. We note,

however, that there are other ways to maximize Φ_{N-k+1} and the next subsection just addresses one such method.

Amend the paragraph beginning at column 69, line 35, as follows:

IV.3.2 Properties of the Lagrangian Relaxed Assignment Problem[.]

For a function evaluation of [$\Phi_{N-k+}1$] Φ_{N-k+1}, we show that an optimal (or suboptimal) solution of this relaxed problem ([3.2]4.8) can be constructed from that of a two-dimensional assignment problem. Then, the nonsmooth characteristics of Φ_{N-k+1} are addressed, followed by a method for computing the function value and a subgradient.

Amend the paragraph beginning at column 69, line 43, as follows:

Evaluation of Φ_{N-k+1}. Define for each (l_k, i_{k+1}) an index $(j_{k+2}, \ldots, j_N) = (j_{k+2}(l_k, i_{k+1}), \ldots, j_N(l_k, i_{k+1}))$ and a new cost function $c^2_{l_k i_{k+1}}$ by

[(3.3)]

$$(j_{k+2}(l_k, i_{k+1}), \ldots, j_N(l_k, i_{k+1})) = \operatorname{argmin}\left\{ \begin{array}{l} c^{N-k+1}_{l_k i_{k+1} i_{k+2} \ldots i_N} + \sum_{p=k+2}^{N} u^p_{i_p} \mid i_p = 0, 1, \ldots, M_p \\ \text{and } p = k+2, \ldots, N \end{array} \right\}, \tag{4.9}$$

$$c^2_{l_k i_{k+1}} = c^{N-k+1}_{l_k i_{k+1} j_{k+2}(l_k, i_{k+1}) \ldots j_N(l_k, i_{k+1})} + \sum_{p=k+2}^{N} u^p_{j_p}(l_k, i_{k+1}) \text{ for } (l_k, i_{k+1}) \neq (0,0),$$

$$c^2_{00} = \sum_{i_{k+1} \ldots i_N} \text{Minimum}\left\{0, c_{00 i_{k+2} \ldots i_N} + \sum_{p=k+2}^{N} u^p_{i_p}\right\},$$

Given an index pair (l_k, i_{k+1}), $(j_{k+2}, \ldots, j_N)$ need not be unique, resulting in the potential generation of several subgradients ([3.11]4.17). Then,

[(3.4)]

$$\hat{\Phi}_{N-k+1}(u^{k+2}, \ldots, u^N) =$$
$$\text{Minimize } \hat{\phi}_{N-k+1}(z^2; u^{k+2}, \ldots, u^N)$$
$$\equiv \sum_{l_k=0}^{L_k} \sum_{i_{k+1}=0}^{M_{k+1}} c^2_{l_k i_{k+1}} z^2_{l_k i_{k+1}} - \sum_{p=k+2}^{N} \sum_{i_p=0}^{M_p} u^p_{i_p} \qquad \underline{(4.10)}$$
$$\text{Subject To } \sum_{i_{k+1}=0}^{M_{k+1}} z^2_{l_k i_{k+1}} = 1,\ l_k = 1, \ldots L_k,$$
$$\sum_{l_k=0}^{L_k} z^2_{l_k i_{k+1}} = 1,\ i_{k+1} = 1, \ldots M_{k+1},$$
$$z^2_{l_k i_{k+1}} \in \{0,1\} \text{ for all } l_k, i_{k+1}.$$

As an aside, two observations are in order. The first is that the search procedure needed for the computation of the relaxed cost coefficients in ([3.3]4.9) is the most computationally intensive part of the entire relaxation algorithm. The second is that a feasible solution z^{N-k+1} of a sparse problem ([3.1]4.7) yields a feasible solution z^2 of ([3.4]4.10) via the construction

$$z^2_{l_k i_{k+1}} = \begin{cases} 1, & \text{if } z^{N-k+1}_{l_k i_{k+1} i_{k+2} \cdots i_N} = 1 \text{ for some } (i_{k+2}, \cdots, i_N), \\ 0, & \text{otherwise.} \end{cases}$$

thus, there are generally solutions other than the one nonzero index solution.

Amend the paragraph beginning at column 70, line 54, as follows:

The following Theorem [3.1]4.1 gives a method for evaluating Φ_{N-k+1} and states that an optimal solution of ([3.2]4.8) can be computed from that of ([3.4]4.10).

Furthermore, if the solution of either of these two problems is ϵ-optimal, then so is the other. The converse is contained in Theorem ([3.2]4.2).

Delete the paragraphs beginning at column 70, line 59, thru column 71, line 67, and replace them with the following new paragraphs:

Theorem 4.1. *Let w^2 be a feasible solution to the two-dimensional assignment problem (4.10) and define w^{N-k+1} by*

$$w^{N-k+1}_{l_k i_{k+1} i_{k+2} \cdots i_N} = w^2_{l_k i_{k+1}} \quad if\,(i_{k+2}, \ldots, i_N) = (j_{k+2}, \ldots, j_N) \text{ and } (l_k, i_{k+1}) \neq (0,0)$$

$$w^{N-k+1}_{l_k i_{k+1} i_{k+2} \cdots N} = 0 \quad if\,(i_{k+2}, \ldots i_N) \neq (j_{k+2}, \ldots, j_N) \text{ and } (l_k, i_{k+1}) \neq (0,0)$$

$$w^{N-k+1}_{00 i_{k+2} \cdots i_N} = 1 \quad if\; c^{N-k+1}_{00 i_{k+2} \cdots i_N} + \sum_{p=k+2}^{N} u^p_{i_p} \leq 0,$$

$$w^{N-k+1}_{00 i_{k+2} \cdots i_N} = 0 \quad if\; c^{N-k+1}_{00 i_{k+2} \cdots i_N} + \sum_{p=k+2}^{N} u^p_{i_p} > 0. \tag{4.11}$$

Then w^{N-k+1} is a feasible solution of the Lagrangian relaxed problem and

$$\phi_{N-k+1}(w^{N-k+1}; u^{k+2}, \ldots, u^N) = \acute{\phi}_{-k-1}(w_2;\; u^{k+2}, \ldots, u^N).$$

If, in addition, w^2 is optimal for the two-dimensional problem, then $w^{N=k+1}$ is an optimal solution of the relaxed problem and $\Phi_{N-k+1}(u^{k+2}, \ldots, u^N) = \acute{\Phi}_{-k+1}(u^{k+2}, \ldots, u^N)$.

Proof. Let w^{N-k+1} and w^2 be as in the hypotheses, and let

$\phi_{N-k+1}(w^{N-k+1}; u^{k+2}, \ldots, u^N)$ and

$\acute{\phi}_{N-k+1}(z^2; u^{k+2}, \ldots, u^N)$

denote the objective function values of (4.8) and (4.10), respectively. Direct verification shows that w^{N-k+1} satisfies the constraints in (4.8) and

$$\phi_{N-k+1}(w^{N-k+1};u^{k+2};\ldots,u^N) = \acute{\phi}_{-k+1}(w^2;u^{k+2},\ldots,u^N).$$

For the remainder of the proof, assume that w^2 is optimal for (4.10). Let x^{N-k+1} satisfy the constraints in (4.8) and define

$$\begin{aligned}
&x^2_{l_k i_{k+1}} = \sum\nolimits_{i_{k+2}\cdots i_N} x^{N-k+1}_{l_k i_{k+1} i_{k+2}\cdots i_N} \text{ for } (l_k, i_{k+1}) \neq (0,0),\\
&\quad x^2_{00} = 1 \text{ if } (l_k, i_{k+1}) = (0,0) \text{ and}\\
&\qquad c^{N-k+1}_{00 i_{k+2}\cdots i_N} + \sum_{p=k+2}^{N} u^p_{i_p} \le 0 \text{ for some } (i_{k+2},\cdots,i_N),\\
&\quad x^2_{00} = 0 \text{ if } (l_k, i_{k+1}) = (0,0) \text{ and}\\
&\qquad c^{N-k+1}_{00 i_{k+2}\cdots i_N} + \sum_{p=k+2}^{N} u^p_{i_p} > 0 \text{ for all } (i_{k+2},\cdots,i_N).
\end{aligned} \tag{4.12}$$

Note that x^2 satisfies the constraints in (4.10) and

$$\begin{aligned}
\phi_{N-k+1}(x^{N-k+1};u^{k+2},\ldots,u^N) &\ge \acute{\phi}_{-k+1}(x^2;u^{k+2},\ldots,u^N)\\
&\ge \acute{\phi}_{N-k+1}(w^2;u^{k+2},\ldots,u^N)\\
&= \phi_{N-k+1}(w^{N-k+1};u^{k+2},\ldots,u^N)
\end{aligned}$$

for any feasible solution x^{N-k+1} of the constraints (4.8). This implies w^{N-k+1} is an optimal solution of (4.8). Next,

$$\Phi_{N-k+1}(u^{k+2},\ldots,u^N) = \acute{\Phi}_{N-k+1}(u^{k+2},\ldots,u^N)$$

follows immediately from

$$\phi_{N-k+1}(w^{N-k+1};u^{k+2},\ldots,u^N) = \acute{\phi}_{N-k+1}(w^2;u^{k+2},\ldots,u^N)$$

for an optimal solution w^2 of (4.10). With the exception of one equality being converted to an inequality, the following theorem is a converse of Theorem 4.1.

Theorem 4.2. *Let w^{N-k+1} be a feasible solution to problem (4.8) and define w^2 by*

$$
\begin{aligned}
& w^2_{l_k i_{k+1}} = \sum_{i_{k+2}\cdots i_N} w^{N-k+1}_{l_k i_{k+1} i_{k+2}\cdots i_N} \text{ for } (l_k, i_{k+1}) \neq (0,0), \\
& w^2_{00} = 1 \text{ if } (l_k, i_{k+1}) = (0,0) \text{ and} \\
& \qquad C^{N-k+1}_{00 i_{k+2}\cdots i_N} + \sum_{p=k+2}^{N} u^p_{i_p} \leq 0 \text{ for some } (i_{k+2}, \cdots, i_N), \\
& w^2_{00} = 0 \text{ if } (l_k, i_{k+1}) = (0,0) \text{ and} \\
& \qquad C^{N-k+1}_{00 i_{k+2}\cdots i_N} + \sum_{p=k+2}^{N} u^p_{i_p} > 0 \text{ for all } (i_{k+2}, \cdots, i_N).
\end{aligned}
\tag{4.13}
$$

Then w^2 a feasible solution of the problem (4.10) and

$$\phi_{N-k+1}(w^{N-k+1}; u^{k+2}, \ldots, u^N) \geq \acute{\phi}_{N-k+1}(w^2; u^{k+2}, \ldots, u^N).$$

If, in addition, w^{N-k+1} is optimal for (4.8), then w^2 is an optimal solution of (4.10),

$$\phi_{N-k+1}(w^{N-k+1}; u^{k+2}, \ldots, u^N) = \acute{\phi}_{N-k+1}(w^2; u^{k+2}, \ldots, u^N) \text{ and}$$

$$\Phi_{N-k+1}(u^{k+2}, \ldots, u^N) = \acute{\Phi}_{N-k+1}(u^{k+2}, \ldots, u^N).$$

Proof: Let w^{N-k+1} and w^2 be as in the hypotheses, and let $\phi_{N-k+1}(w^{N-k+1}; u^{k+2}, \ldots, u^N)$ and $\acute{\phi}_{N-k+1}(w^2; u^{k+2}, \ldots, u^N)$ denote the objective function values of (4.8) and (4.10), respectively. Direct verification shows that w^2 satisfies the constraints in (4.10) and

$$\phi_{N-k+1}(w^{N-k+1}; u^{k+2}, \ldots, u^N) \geq \acute{\phi}_{N-k+1}(w^2; u^{k+2}, \ldots, u^N).$$

For the remainder of the proof, assume that w^{N-k+1} is optimal for (4.8) and construct w^2 as above. Using Theorem 4.1, construct $\overline{w}^{N-k+1}$ by replacing w^{N-k+1} in (4.11) with $\overline{w}^{N-k+1}$. Then, from that theorem

$$\Phi_{N-k+1}(\overline{w}^{N-k+1};u^{k+2},\ldots,u^N) = \acute{\phi}_{N-k+1}(w^2;u^{k+2},\ldots,u^N)$$
$$\leq \Phi_{N-k+1}(w^{N-k+1};u^{k+2},\ldots,u^N).$$

Optimality of w^{N-k+1} then implies the last inequality is in fact an equality. This proves the theorem.

Amend the paragraph beginning at column 72, line 1, as follows:

The Nonsmooth Optimization Problem. Next, we address the nonsmooth properties of the function [Φ_{N-k+1}YU^{k+2}U$^n\Delta$] $\Phi_{N-k+1}(u^{k+2},\ldots,u^N)$ as explained in the following theorem.

Amend the paragraph beginning at column 72, line 4, as follows:

[Theorem 3.4] **Theorem 4.3.** (G.L. Nemhauser and L.A. Wolsey. Integer and Combinatorial Optimization, Section II.3.6. *Wiley, New York, NY, 1988).* Let [Φ_{N-k+1}YnΔ Φ_{N-k+1}Y^{N-k+1}] Φ_{N-k+1} be as defined in (4.7), let $V_{N-k+1}(z^{N-k+1})$ be the [object]objective function value of the (N-k+1)-dimensional assignment problem in [equation (3.1)] (4.7) corresponding to a feasible solution z^{N-k+1} of the constraints in [(3.1)] (4.7), and let
$\bar{z}^{N-k+1}$ be an optimal solution of [(3.1)] (4.7). Then, [Φ_{N-k+1} YU^{k+2}U$^n\Delta$] $\Phi_{N-k+1}(u^{k+2},\ldots,u^N)$ is piecewise affine, concave and continuous in [{U^{k+2}U^N] $\{u^{k+2},\ldots,u^N\}$ and

$$\Phi_{N-k+1}(u^{k+2},\ldots,u^N) \leq V_{N-k+1}(\bar{z}^{N-k+1}) \leq V_{N-k+1}(z^{N-k+1}). \tag{4.14}$$

Most of the algorithms for nonsmooth optimization are based on generalized gradients, called *subgradients*, given by the following definition for a concave function.

Amend the paragraph beginning at column 72, line 16, as follows:

Definition [3.5]4.1. *At* $u=(u^{k+2},\ldots,u^N)$ *the set* $\partial\Phi_{N-k+1}(u)$ *is called a subdifferential of* Φ_{N-k+1} *and is defined for the concave function* $\Phi_{N-k+1}(u)$ *by*

$$\partial\Phi_{N-k+1}(u)=\{g\in\mathbb{R}^{M_{k+2}+1}\times\cdots\times\mathbb{R}^{M_N+1}\,|\,\Phi_{N-k+1}(w)-\Phi_{N-k+1}(u)\le g^T(w-u)\text{ for all } w\in\mathbb{R}^{M_{k+2}+1}\times\cdots\times\mathbb{R}^{M_N+1}\},\tag{4.15}$$

where $g_0^p=w_0^p=u_0^p=0$ *are all permanently fixed. (Recall that these were used for notational conveience only.) A vector* $g\in\partial\Phi_{N-k+1}$ *is called a subgradient.*

Amend the paragraph beginning at column 72, line 25, as follows:

There is a large literature on such problems, e.g., (J.-B. Hiriart-Urruty and C. Lemarechal. *Concex Analysis and Minimization Algorithms I& II.* Springer-Verlag, Berlin, 1993; K.C. Kiwiel. Methods of descent for nondifferentiable optimization. In *Lecture Notes in Mathematics 1133, A. Dold and B. Eckmann, eds. Springer-Verlag, Berlin,* 1985; C.Lemarechal and R. Mifflin. *Nonsmooth Optimization.* Pergamon Press, Oxford, UK, 1978; B.T. Polyak. Subgradient method: A survey of Soviet research. In C. Lemarechal and R. Mifflin, editors, *Nonsmooth Optimization,* pages 5-29, N.Y., 1978. Pergamon Press.; H. Schramm and J. Zowe. A version of the bundle idea for minimizing a nonsmooth function: Conceptual idea, convergence analysis, numerical results. *SIAM Journal on Optimization,* 2, No.1:121-152, February, 1992; N.Z. Shor. *Minimization Methods for Non-Differentiable Functions.* Springer-Verlag, Nem York, 1985; P.Wolfe. A method of

conjugate subgradients for minimizing nondifferentiable functions. *Mathematical Programming Study,* 3:145-173, 1975), and we have tried a variety of methods including subgradient methods (N.Z. Shor. *Minimization Methods for Non-Differentiable Functions.* Springer-Verlag, New York, 1985) and bundle methods (J.-B. Hiriart-Urruty and C. Lemarechal. *Convex Analysis and Minimization Algorithms I& II.* Springer-Verlag, Berlin, 1993; K.C. Kiwiel. Methods of descent for nondifferentiable optimization. In *Lecture Notes in Mathematics 1133, A. Dold and B.Eckmann, eds. Springer-Verlag,* Berlin, 1985). Of these, we have determined that for a fixed number of nonsmooth iterations (e.g., twenty), the bundle trust method of Schramm and Zowe (H.Schramm and J. Zowe. A version of the bundle idea for minimizing a nonsmooth function: Conceptual idea, convergence analysis, numerical results. *SIAM Journal on Optimization,* 2, No. 1:121-152, February, 1992) provides excellent quality solutions with the fewest number of function and subgradient evaluations, and is therefore our currently recommended method.

Amend the paragraph beginning at column 72, line 33, as follows:

An Algorithm for Computing Φ_{N-k+1} and a Subgradient. Most current software for maximizing the concave function Φ_{N-k+1} requires the value of the function and a subgradient at a point $(u^{k+2},...,u^{N})$. Based on the previous two sections, this can be summarized as follows.

1. Starting with problem ([3.1]4.7), form the relaxed problem ([3.2]4.8);

2. To solve ([3.2]4.8) optimally, defined the two-dimensional assignment problem ([3.4]4.10) via the transformation ([3.3]4.9);

3. Solve the two-dimensional problem ([3.4]4.10) optimally;

4. Reconstruct the optimal solution, say $\hat{w}^{N-k+1}$, of ([3.2]4.8) via equation ([3.5]4.9) as in Theorem [3.1]4.1;

5. Compute the function

$$\Phi_{N-k+1}(u^{k+2}, \ldots, u^N) \equiv \phi_{N-k+1}(\hat{w}^{N-k+1}; u^{k+2}, \ldots, u^N) \equiv \sum_{1_k i_{k+1}, \cdots i_N} c^{N-k+1}_{1_k i_{k+1} \cdots i_N} \hat{w}^{N-k+1}_{1_k i_{k+1} \cdots i_N} + \sum_{p=k+2}^{N} \sum_{i_p=0}^{M_p} u^p_{i_p} \left[\sum_{1_k i_{k+1} \cdots i_{p-1} i_{p+1} \cdots i_N} \hat{w}^{N-k+1}_{1_k i_{k+1} \cdots i_N} - 1 \right]; \qquad [(3.10)]\ (4.16)$$

A subgradient is given by

[(3.11)]

$$g^p_{i_p} = \frac{\partial \Phi_{N-k+1}(u^{k+2}, \ldots, u^N)}{\partial u^p_{i_p}} = \left[\sum_{1_k i_{k+1} \cdots i_{p-1} i_{p+1} \cdots i_N} \hat{w}^{N-k+1}_{1_k i_{k+1} \cdots i_N} - 1 \right] \quad \text{for } i_p = 1, \ldots, M_p \text{ and } p = k+2, \ldots, N. \qquad (4.17)$$

Several remarks are in order. First, the optimal solution of the minimization problem ([3.2]4.8) is required before one can remove the minimum sign, replace z^{N-k+1} by the minimizer $\hat{w}^{N-k+1}$ and differentiate with respect to $u^p_{i_p}$ to obtain a subgradient as in ([3.11]4.17). If $\hat{w}^{N-k+1}$ is an approximate solution of ([3.2]4.8), then the subgradient and function

values are only approximate with [accruacy]accuracy depending on that of $\hat{w}^{N-k+1}$. Although one can evaluate the sums in ([3.10]4.16) and ([3.11]4.17) in a [straightforward]straight forward manner, another method is based on the following observation. Given the multiplier vector $(u_{k+2}, \ldots, u_N)$, let $\left\{(l_k(l_{k+1}), i_{k+1}(l_{k+1}))\right\}_{l_{k+1}=0}^{L_{k+1}}$ be an enumeration of indices $\left\{l_k, i_{k+1}\right\}$ of w^2 (or the first 2 indices of w^{N-k+1} constructed in equation [3.5]4.9)) such that $w^2_{l_k(l_{k+1}), i_{k+1}(l_{k+1})} = 1$ for $(l_k(l_{k+1}), i_{k+1}(l_{k+1})) \neq (0,0)$
and $(l_k(l_{k+1}), i_{k+1}(l_{k+1})) = (0,0)$ for $l_{k+1}=0$ regardless of whether
$w^2_{00}=1$ or not. (The latter can only improve the recovered feasible solution.) The evaluation of the bracketed quantity in ([3.11]4.17) for a specific $i_p \geq 1$ and [p=k+2]$p = k+2, \ldots, N$ is one minus the number of times the index value i_p appears in the $(p-k+1)^{th}$ position of the ([N-k+1)-tuple in the list $\left\{l_k(l_{k+1}), i_{k+1}(l_{k+1}), j_{k+2}, \ldots, j_N\right\}_{l_{k+1}=0}^{L_{k+1}}$.

Amend the paragraph beginning at column 73, line 19, as follows:

Finally, we have presented a method for computing one subgradient. If $\hat{w}^{N-k+1}$ is the unique optimal solution of ([3.2]4.8), then $\Phi_{N-k+1}(u)$ is differentiable at u, g is just the gradient at u, and the subdifferential $\partial\Phi_{N-k+1}(u)=\{g\}$ is a singleton. If, corresponding to u, $\hat{w}^{N-k+1}$ is not unique, then there are finitely many such solutions, say $\{\hat{w}^{N-k+1}(1), \ldots, \hat{w}^{N-k+1}(m)\}$ with respective subgradients $\{g(1), \ldots, g(m)\}$, the subdifferential $\partial\Phi(u)$ is the convex hull of $\{g(1), \ldots, g(m)\}$ (J.L. Goffin. On convergence rates of subgradient optimization methods, mathematical programming. *Mathematical Programming*, 13:329-347, 1977). These nonunique

solutions arise in two ways. First, if the optimal solution of the [two dimensional]two-dimensional assignment problem is not unique, then one can [general]generate all optimal solutions of the [two dimensional]two-dimensional assignment problem ([3.4]4.10). Corresponding to each of these solutions and to each index pair (l_k, i_{k+1}) in each solution, compute the indices $(j_{k+2}, \ldots, j_N)$ in ([3.3]4.9). If these j's are not unique, then we can enumerate all the possible optimal solutions $\hat{w}^{N-k+1}$ of ([3.2]4.8). Given these solutions, one can then compute the corresponding subgradients from ([3.11]4.17).

Amend the paragraph beginning at column 73, line 38, as follows:

In most nonsmooth optimization algorithms, one uses an [epsilon-subdifferential] ϵ-subdifferential.

Amend the paragraph beginning at column 73, line 40, as follows:

Definition [3.x]4.2. *At $u = (u^{k+2}, \ldots, u^N)$ the set $\partial_\epsilon \Phi_{N-k+1}(u)$ is called an [epsilon subdifferential] ϵ-subdifferential of Φ_{N-k+1} and is defined for the concave function $\Phi_{N-k+1}(u)$ [(u)] by*

$$\partial_\epsilon \Phi_{N-k+1}(u) = \{ g \in \mathbb{R}^{M_{k+2}+1} \times \ldots \times \mathbb{R}^{M_N+1} \mid \Phi_{N-k+1}(w) - \Phi_{N-k+1}(u) \leq g^T(w-u) + \epsilon \text{ for all } w \in \mathbb{R}^{M_{k-2}+1} \times \ldots \times \mathbb{R}^{M_N+1} \},$$

where $g_0^p = w_0^p = u_0^p = 0$ are all permanently fixed. (Recall that these were used for notational convenicence only.) A vector $g \in \partial_\epsilon \Phi_{N-k+1}(u)$ is called [a subgradient] an ϵ-subgradient.

Delete the paragraph beginning at column 73, line 49.

Amend the paragraph beginning at column 73, line 50, as follows:

IV.3.3 The Recovery Procedure.

The next objective is to explain a *recovery procedure*, i.e., given a feasible (optimal or suboptimal) solution [ω^2 $\gamma\Delta\gamma\omega^{N-k+1}$ $\gamma\Delta$] w^2 of (4.10) (or w^{N-k+1} of (4.8) constructed via Theorem [3.1]4.1), generate a feasible solution z^{N-k+1} of equation ([3.1]4.7) which is close to [ω^2] w^2 in a sense to be specified. We *first assume that no variables in* ([3.1]4.7) *are [presassigned] preassigned to zero; this assumption will be removed shortly.* The difficulty with the solution [ω^{N-k+1}] w^{N-k+1} in Theorem [3.1]4.1 is that it need not satisfy the last *(N-k-1)* sets of constraints in ([3.1]4.7). (Note, however, that if [ω^2]w^2 is an optimal solution ([3.4]4.10) and [ω^{N-k+1}]w^{N-k+1}, as constructed in Theorem 4.1, satisfies the relaxed constraints, then [ω^{N-k+1} $\gamma\Delta\Delta\omega$]w^{N-k+1} is optimal for (4.7).) The recovery [proceudre]procedure described here is designed to preserve the 0-1 character of the solution [ω^2 $\gamma\Delta$]w^2 of (4.10) as far as possible: If [$\omega^2_{1_k i_{k+1}}=1$ *and* $1_k\neq 0$ *or* $i_{k+1}\neq 0$], $w^2_{1_k i_{k+1}}=1$ *and* $1_k\neq 0$ *or* $i_{k+1}\neq 0$ the corresponding feasible solution z^{N-k+1} of ([3.1]4.7) is construed so that $z^{N-k+1}_{1_k i_{k+1}\cdots i_N}=1$ for some $(i_{k+2},\ldots,i_N)$. By this reasoning, variables of the form $z^{N-k+1}_{00 i_{k+2}\cdots i_N}$ can be assigned a value of one in the recovery problem only if [$\omega^2_{00}=1$] $w^2_{00}=1$. However, variables $z^{N-k+1}_{00 i_{k+2}\cdots i_N}$ will be treated differently in the recovery procedure in that they can be assigned either zero or one independent of the value [of] [$\omega^2_{00}=1$] w^2_{00}. This

increases the feasible set of the recovery problem, leading to a potentially better solution.

Amend the paragraph beginning at column 74, line 4, as follows:

Let $\left\{(l_k(l_{k+1}), i_{k+1}(l_{k+1})\right\}_{l_{k+1}=0}^{L_{k+1}}$ be an enumeration of indices $\{l_k, i_{k+1}\}$ of $\underline{w^2}$ <u>(or the first 2 indices of</u> [$\omega^2 \gamma\ \omega^{N-k+1}$] $\underline{w^{N-k+1}}$ <u>constructed in</u> equation [(5)] <u>(4.11)</u>) such that

$$[\omega^2_{l_k(l_{k+1}),\, i_{k+1}(l_{k+1})} = 1]$$
$$\underline{w^2_{l_k(l_{k+1}),\, i_{k+1}(l_{k+1})} = 1} \quad \text{for}$$
$$(l_k(l_{k+1}), i_{k+1}(l_{k+1})) \neq (0,0) \quad \text{and}$$
$$(l_k(l_{k+1}), i_{k+1}(l_{k+1})) = (0,0) \quad \text{for } l_{k+1}=0$$

regardless of whether [$\omega^2_{00}=1$] $\underline{w^2_{00}=1}$ or not. To define the [N-k dimensional] <u>*(N-k)*-dimensional</u> assignment problem that [resores] <u>restores</u> feasibility, let

[(3.12)]

$$c^{N-1}_{l_2 i_3 \ldots i_N} = c^{N}_{l_i(l_2) i_2(l_2) i_3 \ldots i_N} \quad \textit{for } k=1$$

$$c^{N-k}_{l_{k+1} i_{k+2} \ldots i_N} = c^{N-k+1}_{l_k(l_{k+1}) i_{k+1}(l_{k+1}) i_{k+2} \ldots i_N} \qquad \underline{(4.18)}$$

$$= c^{N}_{i_1(l_{2 \ldots k+1}) i_2(l_{2 \ldots k+1}) i_3(l_{3 \ldots k+1}) \ldots i_k(l_{kk+1}) i_{k+1}(l_{k+1}) i_{k+2} \ldots i_N}$$

$$\textit{for } k \geq 2 \textit{ and } l_k = 0, \ldots, L_k,$$

where

$$l_{m \ldots k+1} = l_m \circ l_{m+1} \circ \ldots \circ l_k(l_{k+1}) \equiv l_m(l_{m+1}(\ldots(l_k(l_{k+1}))\ldots)) \qquad \underline{(4.19)}$$

for $m = 2, \ldots k+1$ and where [o] <u>"∘"</u> denotes function composition. For example, $l_{23} = l_2(l_3)$ and $l_{234} = l_2 \circ l_3(l_4) = l_2(l_3(l_4))$.

Amend the paragraph beginning at column 74, line 24, as follows:

Then the *(N-k)*-dimensional assignment problem that restores feasibility is [(3.14)]

$$\text{Minimize} \quad \sum_{l_{k+1} i_{k+2} \ldots i_N} c^{N-k}_{l_{k+1} i_{k+2} \ldots i_N} z^{N-k}_{l_{k+1} i_{k+2} \ldots i_N}$$

$$\text{Subject To} \quad \sum_{i_{k+2} \ldots i_N} z^{N-k}_{l_{k+1} i_{k+2} \ldots i_N} = 1, \; l_{k+1} = 1, \ldots, L_{k+1},$$

$$\sum_{i_{k+1} i_{k+3} \ldots i_N} z^{N-k}_{l_{k+1} i_{k+2} \ldots i_N} = 1, \; i_{k+2} = 1, \ldots, M_{k+2}, \qquad \underline{(4.20)}$$

$$\sum_{l_k i_{k+1} \ldots i_{p-1} i_{p+1} \ldots i_N} z^{N-k}_{l_{k+1} i_{k+2} \ldots i_N} = 1$$

$$\text{for } i_p = 1, \ldots, M_p \text{ and } p = k+3, \ldots, N-1,$$

$$\sum_{l_{k+1} i_{k+2} \ldots i_{N-1}} z^{N-k}_{l_{k+1} i_{k+2} \ldots i_N} = 1, \; i_N = 1, \ldots M_N,$$

$$z^{N-k}_{l_{k+1} i_{k+2} \ldots i_N} \in \{0,1\} \text{ for all } l_{k+1}, i_{k+2}, \ldots, i_N.$$

Let Y be an optimal or feasible solution to this [N - k]<u>*(N-k)-*</u> dimensional assignment problem. The recovered feasible solution z^N is defined by [(3.14)]

$$z^N_{i_1 i_2 i_3 \ldots i_N} = \begin{cases} 1, & \text{if } i_1 = i_1(l_{2\ldots k+1}),\; i_2 = i_2(l_{2\ldots k+1}), \\ & \quad i_3 = i_3(l_{2\ldots k+1}), \ldots,\; i_k = i_k(l_{kk+1}), \\ & \quad i_{k+1} = i_{k+1}(l_{k+1}) \text{ and } Y_{l_{k+1} i_{k+2} \ldots i_N} = 1, \\ 0, & \text{otherwise.} \end{cases} \qquad \underline{(4.21)}$$

Said in a different way, the recovered [reasible]<u>feasible</u> solution [z^N]<u>z^N</u> corresponding to the multiplier set $\{u^{k+2}, \ldots, u^N\}$ is then defined by

$$z^{N}_{i_1(l_{2\cdots k+1})\, i_2(l_{2\cdots k+1})\, i_3(l_{3\cdots k+1})\cdots i_k(l_{kk+1})\, i_{k+1}(l_{k+1})\, i_{k+2}\cdots i_N} = \begin{cases} 1, & \text{if } Y_{l_{k+1} i_{k+2}\cdots i_N} = 1, \\ 0, & \text{otherwise,} \end{cases}$$

where $l_{m\cdots k+1}$ is defined in ([3.13]<u>4.19</u>) and [∘]<u>"∘"</u> denotes function composition.

Amend the paragraph beginning at column 75, line 1, as follows:

The next objective is to remove the a<u>s</u>sumption that all cost coefficients are defined and all zero-one variables are free to be assigned. We first note that the above construction of a reduced dimension assignment problem ([3.14]<u>4.20</u>) is valid as long as all cost coefficients c^{N-k} are defined and all zero-one variables in z^{N-k} are free to be assigned. Modifications are necessary for *sparse* problems. If the cost coefficient $c^{N-k+1}_{l_k(l_{k+1})\, i_{k+1}(l_{k+1})\, i_{k+2}\cdots i_N}$ is well-defined and the zero-one variable $z^{N-k+1}_{l_k(l_{k+1})\, i_{k+1}(l_{k+1})\, i_{k+2}\cdots i_N}$ is free to be assigned to zero or one, then define $c^{N-k}_{l_{k+1} i_{k+2}\cdots i_N} = c^{N-k+1}_{l_k(l_{k+1})\, i_{k+1}(l_{k+1})\, i_{k+2}\cdots i_N}$ as in ([3.12]<u>4.18</u>) with $z^{N-k}_{l_{k+1} i_{k+2}\cdots i_N}$ being free to be assigned. Otherwise, $z^{N-k}_{l_{k+1} i_{k+2}\cdots i_N}$ is preassigned to zero or the corresponding arc is not allowed in the feasible set of arcs. To ensure that a feasible solution exists, we now only need ensure that the variables $z^{N-k}_{l_{k+1}0\cdots 0}$ are free to be assigned for $l_{k+1}=0,1,\ldots,L_{k+1}$ with the corresponding cost coefficients being well-defined. (Recall that all variables of the form $z^{N-k+1}_{l_k 0\cdots 0}$ and $z^{N-k+1}_{0\cdots 0 i_p 0\cdots 0}$ are free (to be assigned) and all coefficients of the form $c^{N-k+1}_{l_k 0\cdots 0}$ and $c^{N-k+1}_{0\cdots 0 i_p 0\cdots 0}$ are well-defined for [$l_k=0$]$\underline{l_k=0}$$,\ldots,L_k$ and [$i_p=0$]$\underline{i_p=0}$$,\ldots,M_p$ for $p=k+1,\ldots,N$.) This is accomplished as follows: If the cost coefficient $c^{N-k+1}_{l_k(l_{k+1})\, i_{k+1}(l_{k+1})\, 0\cdots 0}$ is well-defined and $z^{N-k+1}_{l_k(l_{k+1})\, i_{k+1}(l_{k+1})\, 0\cdots 0}$ is free, then define $c^{N-k}_{l_{k+1}0\cdots 0} = c^{N-k+1}_{l_k(l_{k+1})\, i_{k+1}(l_{k+1})\, 0\cdots 0}$ with $z^{N-k}_{l_{k+1}0\cdots 0}$ being free. Otherwise, since all variables of

the form $z^{N-k+1}_{l_k 0\cdots 0}$ and $z^{N-k+1}_{0 i_{k+1} 0\cdots 0}$ are free to be assigned with the corresponding costs being well-defined, set $c^{N-k}_{l_{k+1} 0\cdots 0}$ = $c^{N-k}_{l_k(l_{k+1}) 00\cdots 0}$ + $c^{N-k+1}_{0 i_{k+1}(l_{k+1}) 0\cdots 0}$, where the first term is omitted if $l_k(l_{k+1})=0$ and the second, if $i_{k+1}(l_{k+1})=0$. For $l_k(l_{k+1})=0$ and $i_{k+1}(l_{k+1})=0$, define $c^{N-k}_{0\cdots 0} = c^{N-k+1}_{0\cdots 0}$.

IV.3.4. The last step k = N-2.

The description of the algorithm ends with [k=N+2]$k = N+2$. The resulting recovery problem defined in ([3.14]4.20) with [k=N-2]$k = N-2$ is

$$\Phi_2 \equiv \mathit{Minimize} \sum_{l_{N-1}=0}^{L_{N-1}} \sum_{i_N=0}^{M_N} c^2_{l_{N-1} i_N} z^2_{l_{N-1} i_N} \equiv V_2(z^2)$$

$$\mathit{Subject\ To} \quad \sum_{i_N=0}^{M_N} z^2_{l_{N-1} i_N} = 1, \; l_{N-1} = 1, \dots, L_{N-1}, \tag{4.22}$$

$$\sum_{l_{N-1}=0}^{L_{N-1}} z^2_{l_{N-1} i_N} = 1, \; l_N = 1, \dots, M_N,$$

$$z^2_{l_{N-1} i_N} \in \{0,1\} \; \mathit{for\ all} \; l_{N-1}, i_N.$$

Let Y be an optimal or feasible solution to this [2-dimensional]two-dimensional assignment problem. The recovered feasible solution z^N is defined by

$$\hat{z}^N_{i_1 i_2 i_3 \cdots i_N} = \begin{cases} 1, & \mathit{if}\ i_1 = i_1(l_{2\cdots N-1}), i_2 = i_2(l_{2\cdots N-1}), i_3 = i_3(l_{3\cdots N-1}), \dots, \\ & i_{N-2} = i_{N-2}(l_{N-2 N-1}), i_{N-1} = i_{N-1}(l_{N-1}) \ \mathit{and}\ Y_{l_{N-1} i_N} = 1 \\ & \mathit{or\ if}\ (l_{N-1}, i_N) = (0,0), \\ 0, & \mathit{otherwise}. \end{cases}$$

([3.18]4.23)

Said in a different way, the recovered feasible solution z^N is then defined by

$$\hat{z}_{i_1(l_{2\cdots N-1})i_2(l_{2\cdots N-1})i_3(l_{2\cdots N-1})\cdots i_{N-2}(l_{N-2N-1})i_{N-1}(l_{N-1})i_N} = \begin{cases} 1, & \text{if } Y_{l_{N-1}i_N} = 1, \\ 0, & \text{otherwise.} \end{cases} \qquad \underline{(4.24)}$$

[(3.19)]

where $l_{m\cdots N-1}$ is defined in ([3.13]<u>4.19</u>) and [∘]<u>"∘"</u> denotes function composition. To complete the description, let $\left\{(l_{N-1}(l_N), i_N(l_N)\right\}_{l_N=0}^{L_N}$ be an enumeration of indices $\{l_{N-1}, i_N\}$ of Y such that $Y_{l_{N-1}(l_N), i_N(l_N)} = 1$ for $\left(l_{N-1}(l_N), i_N(l_N)\right) \neq (0,0)$ and $\left(l_{N-1}(l_N), i_N(l_N)\right) = (0,0)$ *for* $l_N = 0$ regardless of whether $Y_{00}=1$ or not. Then the recovered feasible solution can be written as

$$\hat{z}^N_{i_1 i_2 i_3 \cdots i_N} = \begin{cases} 1, & \text{if } i_1 = i_1(l_{2\cdots N}), i_2 = i_2(l_{2\cdots N}), i_3 = i_3(l_{3\cdots N}), \ldots, \\ & \quad i_{N-1} = i_{N-1}(l_{N-1N}), i_N = i_N(l_N); \\ 0, & \text{otherwise.} \end{cases} \qquad \underline{(4.25)}$$

Amend the paragraph beginning at column 76, line 21, as follows:

<u>IV.3.5.</u> The Upper and Lower [Bound.]<u>Bounds</u>

The upper bound on the feasible solution is given by

$$V_N(\hat{z}^N) = \sum_{i_1 \cdots i_N} c^N_{i_1 \cdots i_N} \hat{z}^N_{i_1 \cdots i_N} \qquad ([3.21]\underline{4.26})$$

and the lower by $[\Phi_N]\underline{\Phi_N}(u^3, \ldots, u^N)$ for any multiplier value $(u^3, \ldots, u^N)$. In particular, we have

$$\Phi_N(u^3, \ldots, u^N) \leq \Phi_N(\overline{u}^3, \ldots, \overline{u}^N) \leq V_N(\overline{z}^N) \leq V_N(\hat{z}^N), \qquad ([3.22]\underline{4.27})$$

where $(u^3, \ldots, u^N)$ is any multiplier value, $(\bar{u}^3, \ldots, \bar{u}^N)$ is a maximizer of $[\Phi_N]\underline{\Phi_N}(u^3, \ldots, u^N)$, $\bar{z}^N$ is an optimal solution of the

multidimensional assignment problem ([3.1]4.7) and $\hat{z}^N$ is the recovered feasible solution defined by ([3.20]4.25). Finally, we conclude with the observation that $V_N(\hat{\underline{z}})=V_2(Y)$ where Y is the optimal solution of ([3.17]4.23) used in the construction of $\hat{\underline{z}}$ in equations ([3.18]4.24)-([3.20]4.26).

Amend the paragraph beginning at column 76, line 36, as follows:

IV.3.6. Reuse of Multipliers[.]

Since the most computationally expensive part of the algorithm occurs in the maximization of Φ_{N-k+1}, the development of algorithms that can make use of hot starts or multipliers close to the optimal are fundamentally important for real-time speed. The purpose of this section is to demonstrate that the multiplier set obtained at stage $k\geq 1$ provide good starting values for those obtained at step $k+1$.

Theorem [3.xx]4.4. *Let $(u^3,\ldots,u^N)$ denote a multiplier set obtained in the [maximum]maximization of $\Phi_N[(](u^3,\ldots,u^N)$. Then this multiplier set satisfies the string of inequalities*

$$\Phi_N(u^3,\ldots,u^N)\leq\Phi_{N-1}(u^4,\ldots,u^N)\leq\ldots\leq\Phi_3(u^N)\leq\Phi_2\leq V_N(\overline{z}), \qquad (4.28)$$

where after the first step in the maximization of Φ_N the multipliers are not changed in the remaining steps. Furthermore, to improve this inequiality, let $(u^{2+k,N-k+1},\ldots,u^{N,N-k+1})$ denote a maximizer of $\Phi_{N-k+1}(u^{k+2},\ldots,u^N)$. Then, we have

$$\Phi_{N-k+1}(u^{k+2},\ldots,u^{N}) \le \Phi_{N-k+1}(u^{2+k,N-k+1},\ldots,u^{N,N-k+1}) \qquad \underline{(4.29)}$$
$$\le \Phi_{N-k}(u^{3+k,N-k+1},\ldots,u^{N,N-k+1})$$
$$\le \Phi_{N-k}(u^{3+k,N-k},\ldots,u^{N,N-k})$$
$$\le \ldots \le \Phi_{3}(u^{N,3}) \le \Phi_{2} \le V_{N}(\hat{z}).$$

[(3.xxx)]

Amend the paragraph beginning at column 76, line 64, as follows:

[Inequalities depend on solution of 2d assignment problem.]

Proof: Suppose we have a value of the multipliers $(u^{k+2},\ldots,u_{N})$ obtained in the maximization of

$$\Phi_{N-k+1}(u^{k+2},\ldots,u^{N}) \equiv Minimize \quad \phi_{N-k+1}(z^{N-k+1};u^{k+2},\ldots,u^{N}) \qquad \underline{(4.30)}$$

$$Subject\ To \quad \sum_{i_{k+1}i_{k+2}\ldots i_{N}} z^{N-k+1}_{l_k i_{k+1} i_{k+2}\ldots i_N}=1,\ l_k=1,\ldots L_k,$$

$$\sum_{l_k i_{k+2}\ldots i_N} z^{N-k+1}_{l_k i_{k+1} i_{k+2}\ldots i_N}=1,\ i_{k+1}=1,\ldots,M_{k+1},$$

where [(3.xx)]

$$\phi_{N-k+1}(z^{N-k+1};u^{k+2},\ldots,u^{N})$$

$$= \sum_{l_k i_{k+1}\ldots i_N} c^{N-k+1}_{l_k i_{k+1}\ldots i_N} z^{N-k+1}_{l_k i_{k+1}\ldots i_N}$$

$$+ \sum_{p=k+2}^{N} \sum_{i_p=0}^{M_p} \left[\sum_{l_k i_{k+1}\ldots i_{p-1} i_{p+1}\ldots i_N} z^{N-k+1}_{l_k i_{k+1}\ldots i_N} - 1 \right] \qquad \underline{(4.31)}$$

$$= \sum_{l_k i_{k+1}\ldots i_N} \left[c^{N-k+1}_{l_k i_{k+1}\ldots i_N} + \sum_{p=k+2}^{N} u^{p}_{i_p} \right] z^{N-k+1}_{l_k i_{k+1}\ldots i_N} - \sum_{p=k+2}^{N} \sum_{i_p=0}^{M_p} u^{p}_{i_p}.$$

Amend the paragraph beginning at column 77, line 24, as follows:

These need not be the maximizer; however, we do assume that we have solved the above minimization problem optimally to evaluate $\Phi_{N-k+1}(u^{k+2},\ldots,u^N)$. Just as in the definition of the [earlier]recovery problem discussed earlier, let $\{(l_k(l_{k+1}), i_{k+1}(l_{k+1}))\}_{l_{k+1}=0}^{L_{k+1}}$ be an enumeration of indices $\{l_k, i_{k+1}[)]\}$ of the optimal solution w^2 of problem ([3.4]4.10) (or the first 2 indices of the solution [$w^{N=k+1}$] w^{N-k+1} of the relaxed problem ([3.2]4.8) constructed in equation ([3.5]4.11) such that $w^2_{l_k(l_{k+1}), i_{k+1}(l_{k+1})}=1$ for $(l_k(l_{k+1}), i_{k+1}(l_{k+1}))\neq(0,0)$ and $(l_k(l_{k+1}), i_{k+1}(l_{k+1}))=(0,0)$ for $l_{k+1}=0$ regardless of whether $w^2_{00}=1$ or not.

Amend the paragraph beginning at column 77, line 34, as follows:

Then, the final value of the objective function can be expressed as

$$\Phi_{N-k+1}(u^{k+2},\ldots,u_N) \equiv$$

$$\text{Minimize } \phi_{N-k+1}(z^{N-k+1}; u^{k+2},\ldots,u^N) \qquad \underline{(4.32)}$$

$$\text{Subject to } \sum_{i_{k+2}\cdots i_N} z^{N-k-1}_{l_k(l_{k+1})i_{k+1}(l_{k+1})i_{k+2}\cdots i_N}=1,\ l_{k+1}=1,\ldots,L_{k+1},$$

where

$$\phi_{n-k+1}(z^{N-k+1}; u^{k+2},\ldots,u^N) = \qquad \underline{(4.33)}[(3.xxx)]$$

$$\sum_{l_{k+1}i_{k+2}\cdots i_N}\left[c^{N-k+1}_{l_k(l_{k+1})i_{k+1}(l_{k+1})i_{k+2}\cdots i_N}+\sum_{p=k+2}^{N} u^p_{i_p}\right]z^{N-k+1}_{l_k(l_{k+1})i_{k+1}(l_{k+1})i_{k+2}\cdots i_N}$$

$$-\sum_{p=k+2}^{N}\sum_{i_p=0}^{M_p} u^p_{i_p}.$$

If now another constrain set, say the $(k+2)^{th}$ set, is added to this problem, one has

$$\tilde{\Phi}_{N=k+1}(u^{k+2},\ldots u^{N}) \equiv Minimize\ \tilde{\phi}_{N-k+1}(z^{N-k+1};u^{k+2},\ldots,u^{N}) \equiv \qquad \underline{(4.34)}$$

$$Minimize \sum_{l_{k+1}i_{k+2}\cdots i_N}\left\{c^{N-k+1}_{l_k(l_{k+1})i_{k+1}(l_{k+1})\cdots i_N}+\sum_{p=k+2}^{N} u^{p}_{i_p}\right\}z^{N-k+1}_{l_k(l_{k+1})i_{k+1}(l_{k+1})\cdots i_N}$$

$$-\sum_{p=k+2}^{N}\sum_{i_p=0}^{M_p} u^{p}_{i_p}$$

$$Subject\ to \sum_{i_{k+2}\cdots i_N} z^{N-k+1}_{l_k(l_{k+1})i_{k+1}(l_{k+1})i_{k+2}\cdots i_N}=1,\ l_{k+1}=1,\ldots,L_{k+1},$$

$$\sum_{l_{k+1}i_{k+3}\cdots i_N} z^{N-k+1}_{l_k(l_{k+1})i_{k+1}(l_{k+1})i_{k+2}\cdots i_N}=1,\ i_{k+2}=1,\ldots,M_{k+2}.$$

Amend the paragraph beginning at column 78, line 8, as follows:

Since the constraint $\sum_{l_{k+1}i_{k+3}\cdots i_N} z^{N-k+1}_{l_k(l_{k+1})i_{k+1}(l_{k+1})i_{k+2}\cdots i_N}=1$ for $i_{k+2}=1,\ldots,M_{k+2}$ is now imposed, the feasible region is smaller, so that one has

$$\begin{aligned}\Phi_{N-k+1}(u^{k+2},\ldots,u^{N}) &\le \tilde{\Phi}_{N-k+1}(u^{k+2},\ldots,u^{N})\\ &\equiv \tilde{\Phi}_{N-k+1}(u^{k+3},\ldots,u^{N})\\ &\equiv \Phi_{N-k+1}(u^{k+3},\ldots,u^{N}),\end{aligned}$$

where the fact that $\tilde{\Phi}_{N=k+1}$ does not depend on the multiplier set u^{k+2} due the added constraint set. Also, the last equivalence follows from the fact that $[\Phi_{N=k+1}]\underline{\Phi}_{N=k+1}(u^{k+3},\ldots,u^{N})$ is the relaxed problem[(3.xx)] (with k replaced by $k+1$) for the recovery problem[(3.xx)] in step k of the above algorithm. Continuing this way, one can compute $(u^{3},\ldots,u^{N})$ at

the first step (k=1), fix them thereafter, and perform no optimization at the subsequent steps to obtain

$$\Phi_N(u^3,\dots,u^N) \le \Phi_{N-1}(u^4,\dots,u^N) \le \dots \le \Phi_3(u^N) \le \Phi_2 \equiv V_N(\hat{z}),$$

where Φ_2 is defined in (<u>4.22</u>).

Amend the paragraph beginning at column 78, line 25, as follows:

To explain how to improve this inequality, let $(u^{2+k,N-k+1},\dots,u^{N,N-k+1})$ denote a maximizer of $\Phi_{N-k+1}(u^{k+2},\dots,u^N)$. Then by the same reasoning one has [(3.xx)]<u>the result</u> as stated in the theorem. [Q.E.D.]

Amend the paragraph beginning at column 78, line 29, as follows:

<u>V.</u> Summary of the Lagrangian Relaxation Problem

Given the multidimensional assignment problem

[(4.1)]

$$\begin{aligned}
&\textit{Minimize} && \sum_{i_1\dots i_N} c_{i_1\dots i_N} z_{i_1\dots i_N} \\
&\textit{Subject To} && \sum_{i_2 i_3\dots i_N} z_{i_1\dots i_N} = 1 \quad (i_1 = 1,\dots M_1), \qquad \underline{(5.1)} \\
& && \sum_{i_1 i_3\dots i_N} z_{i_1\dots i_N} = 1 \quad (i_2 = 1,\dots M_2), \\
& && \sum_{i_1\dots i_{p-1} i_{p+1}\dots i_N} z_{i_1\dots i_N} = 1 \quad (i_p = 1,\dots M_p \textit{ and } p = 2,\dots,N-1), \\
& && \sum_{i_1 i_2\dots i_{N-1}} z_{i_1\dots i_N} = 1, \quad (i_N = 1,\dots M_N), \\
& && z_{i_1,\dots i_N} \in \{0,1\} \textit{ for all } i_1,\dots i_N,
\end{aligned}$$

where $c^N_{0\dots0}$ is arbitrarily defined to be zero and is included for notational convenience and where the superscript N on both c and z is not an exponent, but denotes the dimension of

the subscripts and the assignment problem as stated in ([2.5]3.5). In relaxed and recovery problems $c^N_{0\cdots0}$ need not be zero! In this problem, we assume that all zero-one variables $z^N_{i_1\cdots i_N}$ with precisely one nonzero index are free to be assigned and that the corresponding cost coefficients are well-defined. (This is a valid assumption in the tracking environment (A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. Cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, R.I.,* 19:169-198, 1995; A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications,* 3:27-57, 1994).) Although not required, these cost coefficients with exactly one nonzero index can be translated to zero by cost shifting (A.B. Poore and N. Rijavec. A lagrangian relaxation algorithm for multidimensional assignment problems arising from multitarget tracking. *SIAM Journal of Optimization,* 3, No. 3:544-563, 1993) without changing the optimal assignment.

Amend the paragraph beginning at column 78, line 61, as follows:

Having explained many of the relaxation features, it is now appropriate to present the Lagrangian relaxation algorithm, which iteratively constructs a feasible solution to the *N*-dimensional assignment problem ([4.1]5.1).

Amend the paragraph beginning at column 78, line 65, as follows:

Algorithm [4.1]5.1 *(Lagrangian Relaxation Algorithm)[.] To construct a high quality feasible solution, [enoted]denoted by $\hat{z}^N$, of the assignment problem ([4.1]5.1), proceed as follows:*

0. *Initialize the multipliers* $(u^{k+2},\dots,u^N)$, *e.g.,* $(u^{k+2},\dots,u^N)=(0,\dots,0)$.

For k-1,…,N-2, do

1. *For the Lagrangian relaxed problem ([3.2]4.8) from the problem ([3.1]4.7) by relaxing the last* (N-k-1) *sets of constraints.*
2. *Use a nonsmooth optimization technique to solve*

[(4.2)]

$$\text{Maximize } \{\phi_{N-k+1} \mid u^p \in \mathbb{R}^{M_p+1} \text{ for } p=k+2,\dots,N \text{ with } u_0^p=0 \text{ being fixed}\}, \qquad (5.2)$$

where $\Phi_{N-k+1}(u^{k+2},\dots,u^N)$ *is defined by equation ([3.2]4.8). The algorithm [following problem (3.9)]in Section IV.3.2 provides one method for computing a function value and a subgradient out of the subdifferential at* $(u^{k+2},\dots,u^N)$, *as required in most nonsmooth optimization techniques.*

3. *Given an approximate or optimal maximizer of ([4.2]5.2), say* $(\hat{u}^{k+2},\cdots,\hat{u}^N)$, *let* [$\hat{w}^2$]$\hat{w}^2$*denote the optimal solution of the two-dimensional assignment problem ([3.4]4.10) corresponding to this maximizer of* $\Phi_{N-k+1}(u^{k+2},\dots,u^N)$.
4. *Formulate the recovery* (N-k)*-dimensional problem ([3.13]4.19), modified as discussed at the end of subsection [(3.3)](IV.3.2) for sparse problems. At this*

stage, $\hat{z}^N$, as defined in ([3.15]4.21), contains the alignment of the indices $\{i_1,\ldots,i_{k+1}\}$.

Formulate the final two-dimensional problem ([3.17]4.22) and complete the final recovered solution $\hat{z}^N$ as in ([3.20]4.25).

Amend the paragraph beginning at column 79, line 30, as follows:

To explain the lower and upper bounds, let Φ_N be as defined in ([3.2]4.8) with $k=1$, let $V_N(z^N)$ be the objective function value of the N-dimensional assignment problem in equation ([2.5]3.5) corresponding to a feasible solution z^N of the constraints in ([2.5]3.5), and let $\bar{z}^{N-k+1}$ be an optimal solution of ([2.5]3.5). Then,

$$\Phi_N(u^3,\ldots,u^N) \leq V_N(\tilde{z}^N) \leq V_N(\hat{z}^N) \qquad ([4.3]5.3)$$

is the desired inequality.

Amend the paragraph beginning at column 79, line 41, as follows:

COMMENTS:

1. Step 2 is the computational part of the algorithm. Evaluating Φ_{N-k+1} and computing a subgradient used in the search procedure requires 99% of the computing time in the algorithm. This part uses [two dimensional] two-dimensional assignment algorithms, a search over a large number of indices, and a nonsmooth optimization algorithm. It is the second part (the search) that consumes 99% of the computational time and this is almost entirely parallelizable.
2. In track maintenance, the warm starts for the initial multipliers for the first step are available. For the

relaxation procedure, initial multipliers are available in step 2 from the prior step in the algorithm.

3. There are many variations on the above algorithm. For example, one can compute a good solution on the first pass (k=1) and not perform the optimization in (2) thereafter. This yields a great solution. Thus one can continue the optimization at the first pass, and immediately compute quality feasible solutions to the problem.

Amend the paragraph beginning at column 79, line 64, as follows:

VI. Lagrangian Relaxation Algorithm for the 3-Dimensional Algorithm. [(ABP)]

In this section, we illustrate the relaxation algorithm for the [three dimensional] three-dimensional assignment problem. Having discussed the general relaxation scheme,

$$
\begin{aligned}
&\textit{Minimize} && \sum_{i_1=0}^{M_1}\sum_{i_2=0}^{M_2}\sum_{i_3=0}^{M_3} c_{i_1 i_2 i_3} z_{i_1 i_2 i_3} \\
&\textit{Subject To} && \sum_{i_2=0}^{M_2}\sum_{i_3=0}^{M_3} z_{i_1 i_2 i_3} = 1,\ i_1 = 1,\dots,M_1, \\
& && \sum_{i_1=0}^{M_1}\sum_{i_3=0}^{M_3} z_{i_1 i_2 i_3} = 1,\ i_2 = 1,\dots,M_2, \qquad ([5.1]\underline{6.1}) \\
& && \sum_{i_1=0}^{M_1}\sum_{i_2=0}^{M_2} z_{i_1 i_2 i_3} = 1,\ i_3 = 1,\dots,M_3, \\
& && z_{i_1 i_2 i_3} \in (0,1)\ \textit{for all}\ i_1 i_2 i_3 .
\end{aligned}
$$

To ensure that a feasible solution of ([5.1]6.1) always exists for a sparse problem, all variables with exactly one nonzero index (i.e., variables of the form $z_{i_1 00}$, $z_{0 i_2 0}$, and $z_{00 i_3}$ for $i_p=1,\dots,M_p$ and $p=1,2,3$ are assumed free to be assigned with the corresponding cost coefficients being well-defined). This assumption is valid in the tracking environment [[**refPoa, **refPob].] (A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. Cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence, R.I.*, 19:169-198, 1995; A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994).

Amend the paragraph beginning at column, 80, line 24, as follows:

VI.1. The Lagrangian Relaxed Assignment Problem[.]

The [three dimensional] three-dimensional assignment problem ([5.1]6.1) has three sets of constraints and one can describe the relaxation by relaxing any of the three sets of constraints, the description here is based on relaxing the last set of constraints. A (M_3+1)-dimensional multiplier vector (i.e., $u^3 \in \mathbb{R}^{M_3+1}$) associated with the 3[th]rd constraint set will be denoted by $u^3 = \left(u_0^3, u_1^3, \dots, u_{M_3}^3\right)^T$ with $u_0^3 \equiv 0$ being fixed for notational convenience. (The zero multiplier $u_0^3 \equiv 0$ is used for notational convenience.) Now, the three-dimensional assignment problem ([5.1]6.1) is relaxed to a two-dimensional assignment problem by incorporating last set of constraints into the objective function via the Lagrangian. Although any

constraint set can be relaxed, we choose the *last* set of constraints for convenience. The *relaxed problem* is

([5.2]6.2)

$$\begin{aligned}
\Phi_3(u^3) \equiv \mathit{Minimize}\ \phi_3(z^3;u^3) &\equiv \sum_{i_1=0}^{M_1}\sum_{i_2=0}^{M_2}\sum_{i_3=0}^{M_3} c^3_{i_1i_2i_3} z^3_{i_1i_2i_3} \\
&\quad + \sum_{i_3=0}^{M_3} u^3_{i_3}\left[\sum_{i_1=0}^{M_1}\sum_{i_2=0}^{M_2} z^3_{i_1i_2i_3} - 1\right] \\
&\equiv \mathit{Minimize} \sum_{i_1=0}^{M_1}\sum_{i_2=0}^{M_2}\sum_{i_3=0}^{M_3}\left[c^3_{i_1i_2i_3} + u^3_{i_3}\right] z^3_{i_1i_2i_3} - \sum_{i_3=0}^{M_3} u^3_{i_3} \\
\mathit{Subject\ to}\ &\sum_{i_2=0}^{M_2}\sum_{i_3=0}^{M_3} z^3_{i_1i_2i_3} = 1,\ i_1 = 1,\dots,M_1, \\
&\sum_{i_1=0}^{M_1}\sum_{i_3=0}^{M_3} z^3_{i_1i_2i_3} = 1,\ i_2 = 1,\dots,M_2.
\end{aligned}$$

Amend the paragraph beginning at column 80, line 57, as follows:

One of the major steps in the algorithm is the maximization of [$\Phi_3\gamma^3\Delta u^3$] $\Phi_3(u^3)$ with respect to the multiplier vector u^3. It turns out that Φ_3 is a concave, continuous, and piecewise affine function of the multiplier vector u^3, so that the maximization of Φ_3 is a problem of nonsmooth optimization. Since many of these algorithms require a function value and a subgradient of [$\Phi_3\gamma^3\Delta$] Φ_3 at any required multiplier value (u^3), we address this problem in the next subsection. We note, however, that there are other ways to maximize Φ_3 and the next subsection addresses but one such method.

Amend the paragraph beginning at column 80, line 65, as follows:

VI.2. Properties Lagrangian Relaxed Assignment Problem[.]

For a function evaluation of [Φ_3] Φ_3, we show that an optimal (or suboptimal) solution of this relaxed problem ([5.2]6.2) can be constructed from that of a two-dimensional assignment problem. Then, the nonsmooth characteristics of [Φ_3] Φ_3 are addressed, followed by a method for computing a subgradient.

Amend the paragraph beginning at column 81, line 4, as follows:

Evaluation of Φ_3. Define for each (i_1, i_2) an index $j_3 = j_3(i_1, i_2)$ and a new cost function $c^2_{i_1 i_2}$ by:

$$j_3 \equiv j_3(i_1, i_2) = arg\ min\left\{c^3_{i_1 i_2 i_3} + u^3_{i_3} \mid i_3 = 0, 1, \ldots, M_3\right\}, \qquad ([5.3]6.3)$$

$$c^2_{i_1 i_2} = c^3_{i_1 i_2 j_3(i_1 i_2)} + u^3_{j_3}\ for\ (i_1, i_2) \neq (0, 0),$$

$$c^2_{00} = \sum_{i_3=0}^{M_3} Minimum\left\{0, c^3_{00 i_3} + u^3_{i_3}\right\}.$$

Given an index pair (i_1, i_2), j_3 need not be unique, resulting in the potential generation of several subgradients ([5.11]6.11). (We further discuss this issue at the end of the Subsection [5.2.3].)

Amend the paragraph beginning at column 81, line 18, as follows:

Now, define

$$\hat{\Phi}_3(u^3) = \mathit{Minimize} \quad \hat{\phi}_3(z^2;u^3) \equiv \sum_{i_1=0}^{M_1}\sum_{i_2=0}^{M_2} c^2_{i_1i_2} z^2_{i_1i_2} - \sum_{i_3=0}^{M_3} u^3_{i_3}.$$

$$\text{Subject To} \quad \sum_{i_2=0}^{M_2} z^2_{i_1i_2} = 1, \; i_1 = 1,\ldots,M_1, \qquad \underline{(6.4)}$$

$$\sum_{i_1=0}^{M_1} z^2_{i_1i_2} = 1, \; i_2 = 1,\ldots,M_2,$$

$$z^2_{i_1i_2} \in \{0,1\} \text{ for all } i_1, i_2.$$

[(5.4)]

As an aside, two observations are in order. The first is that the search procedure needed for the computation of the relaxed cost coefficients in ([5.3]6.3) is the most computationally intensive part of the entire relaxation algorithm. The second is that a feasible solution $[z^3]\underline{z^3}$ of a sparse problem ([5.1]6.1) yields a feasible solution z^2 of ([5.4]6.4) via the construction

$$z^2_{i_1i_2} = \begin{cases} 1, & \text{if } z^3_{i_1i_2,i_3} = 1 \text{ for some } (i_1,i_2,i_3), \\ 0, & \text{otherwise.} \end{cases}$$

Thus the two-dimensional assignment problem ([5.4]6.4) has feasible solutions other than those with exactly one nonzero index if the original problem ([5.1]6.1) does.

Amend the paragraph beginning at column 81, line 47, as follows:

The following Theorem [5.1]6.1 states that an optimal solution of ([5.2]6.2) can be computed from that of ([5.4]6.4). Furthermore, if the solution of either of these two problems is ϵ-optimal, then so is the other. The converse is contained in Theorem [5.2]6.2.

Amend the paragraph beginning at column 81, line 51, as follows:

Theorem [5.1]6.1. *Let* $[\omega^2]w^2$ *be a feasible solution to the [two dimensional] two-dimensional assignment problem ([5.4]6.4) and define* w^3 *by*

$$
\begin{aligned}
w^3_{i_1 i_2 i_3} &= w^2_{i_1 i_2} && \text{if } i_3 = j_3 \text{ and } (i_1, i_2) \neq (0,0), \\
w^3_{i_1 i_2 i_3} &= 0 && \text{if } i_3 \neq j_3 \text{ and } (i_1, i_2) \neq (0,0), \\
w^3_{00 i_3} &= 1 && \text{if } c^3_{00 i_3} + u^3_{i_3} \leq 0, \\
w^3_{00 i_3} &= 0 && \text{if } c^3_{00 i_3} + u^3_{i_3} > 0.
\end{aligned}
\qquad ([5.5]6.5)
$$

Then w^3 *is a feasible solution of the Lagrangian relaxed problem (6.2) and* $[\Phi\Upsilon^3\ {}^3\Delta\ \Phi\Upsilon^2\ {}^3\Delta\ \in\ \in^2]$ $\phi_3(w^3;u^3) = \hat{\phi}_3(w^2;u^3)$. *If, in addition,* w^2 *is optimal for the [two dimensional] two-dimensional problem, then* w^3 *is an optimal solution of the relaxed problem and* $\Phi_3(u^3) = \hat{\Phi}_3(u^3)$.

Amend the paragraph beginning at column 81, line 63, as follows:

With the exception of one equality being converted to an inequality, the following theorem is a converse of Theorem [5.1]6.1.

Amend the paragraph beginning at column 81, line 66, as follows:

Theorem [5.2]6.2. *Let* w^2 *be a feasible solution to problem ([5.2]6.2) and define* w^2 *by*

([5.7]<u>6.6</u>)

$$w^2_{i_1 i_2} = \sum_{i_3=0}^{M_3} w^3_{i_1 i_2 i_3} \text{ for } (i_1, i_2) \neq (0,0),$$

$$w^2_{00} = 0 \text{ if } (i_1, i_2) = (0,0) \text{ and } c^3_{00 i_3} + u^3_{i_3} > 0 \text{ for all } i_3,$$

$$w^2_{00} = 1 \text{ if } (i_1, i_2) = (0,0) \text{ and } c^3_{00 i_3} + u^3_{i_3} \leq 0 \text{ for some } i_3.$$

Then w^2 is a feasible solution of the problem ([5.4]<u>6.4</u>) and $\phi_3(w^3; u^3) \geq \hat{\phi}_3(w^2; u^3)$. If, in addition, w^3 is optimal for ([5.2]<u>6.2</u>), then w^2 is an optimal solution of ([5.4]<u>6.4</u>), $\phi_3(w^3; u^3) = \hat{\phi}_3(w^2; u^3)$ and $\Phi_3(u^3) = \hat{\Phi}_3(u^3)$.

Amend the paragraph beginning at column, 82, line 16, as follows:

Given [u^3]<u>u^3</u> the problem of computing [$\Phi^3\Upsilon^3\Delta$] <u>$\Phi_3(u^3)$ and a</u> subgradient of [$\Phi^3\Upsilon^3\Delta$] <u>$\Phi_3(u^3)$</u> can be summarized as follows.

1. Starting with problem ([5.1]<u>6.1</u>), form the relaxed problem ([5.2]<u>6.2</u>)<u>.</u>
2. To solve ([5.2]<u>6.2</u>) optimally, define the [two dimensional] <u>two-dimensional</u> assignment problem ([5.4]<u>6.4</u>) via the transformation ([5.3]<u>6.3</u>)<u>.</u>
3. Solve the two-dimensional problem ([5.4]<u>6.4</u>) optimally.
4. Reconstruct the optimal solution, say [$\hat{w}$]$^{[3]}$<u>$\hat{w}^3$</u>, of ([5.2]<u>6.2</u>) via equation ([5.6]<u>6.6</u>) as in Theorem [5.1]<u>6.1</u>.
5. Then

[(5.10)]

$$\Phi_3(u^3) \equiv \phi_3(\hat{w}^3; u^3) = \sum_{i_1=0}^{M_1} \sum_{i_2=0}^{M_2} \sum_{i_3=0}^{M_3} c^3_{i_1 i_2 i_3} \hat{w}^3_{i_1 i_2 i_3} + \sum_{i_3=0}^{M_3} u^3_{i_3} \left[\sum_{i_1=0}^{M_1} \sum_{i_2=0}^{M_2} \hat{w}^3_{i_1 i_2 i_3} - 1 \right]. \qquad \underline{(6.7)}$$

6. A subgradient is given by substituting[$\hat{W}^3$] $\hat{w}^3$ into the objective function ([5.2]6.2), erasing the minimum sign, and taking the partial with respect to $u^3_{i_3}$. The result is [(5.11)]

$$g^3_{i_3} = \frac{\partial \Phi_3(u^3)}{\partial u^3_{i_3}} = \left[\sum_{i_1=0}^{M_1} \sum_{i_2=0}^{M_2} \hat{w}^3_{i_1 i_2 i_3} - 1 \right] \quad for \ i_3 = 1, \ldots, M_3. \qquad \underline{(6.8)}$$

Amend the paragraph beginning at column, 82, line 44, as follows:

VI.3. The Recovery Procedure[.]

The next objective is to explain a *recovery procedure*, i.e., given a feasible (optimal or suboptimal) solution w^2 of ([5.4]6.4) (or w^3 of ([5.2]6.2) constructed in Theorem [5.1]6.1), generate a feasible solution [z^3]z^3 of equation ([5.1]6.1) which is close to w^2 in a sense to be specified. We first assume that no variables in ([5.1]6.1) are preassigned to zero; this assumption will be removed shortly. The difficulty with the solution w^3 in Theorem [5.1]6.1 is that it need not satisfy the third set of constraints in ([5.1]6.1). (Note, however, that if w^2 is an optimal solution for ([5.4]6.4) and[ ] w^3, as constructed in Theorem [5.1]6.1, satisfies the relaxed constraints, then w^3 is optimal for ([5.1]6.1).) The recovery procedure described here is designed to preserve the 0-1

character of the solution w^2 of ([5.4]6.4) as far as possible: If $w^2_{i_1 i_2}=1$ and $i_1 \neq 0$ or $i_2 \neq 0$, the corresponding feasible solution z^3 of ([5.1]6.1) is constructed so that $z^3_{i_1 i_2 i_3}=1$ for some (i_1, i_2, i_3). By this reasoning, variables of the form $z^3_{00i_3}$ [,] can be assigned a value of one in the recovery problem only if $w^2_{00}=1$. However, variables $z^3_{00i_3}$ will be treated differently in the recovery procedure in that they can be assigned either_ zero or one independent of the value w^2_{00}. This increases the feasible set of the recovery problem, leading to a potentially better solution.

Amend the paragraph beginning at column 82, line 66, as follows:

Let $\left\{(i_1(l_2), i_2(l_2)\right\}_{l_2=0}^{L_1}$ be an enumeration of indices $\{i_1, i_2\}$ of w^2 (or the first 2 indices of w^3 constructed in equation [(5)](6.5) such that $w^2_{i_1(l_2), i_2(l_2)}=1$ for $\left(i_1(l_2), i_2(l_2)\right) \neq (0,0)$ and $\left(i_1(l_2), i_2(l_2)\right)=(0,0)$ for $l_2=0$ regardless of whether $w^2_{00}=1$ or not. (The latter can only improve the quality of the feasible solution.)

Amend the paragraph beginning at column, 83, line 5, as follows:

Next to define the two-dimensional assignment problem that restores feasibility, let

$$c^2_{l_2 i_3} = c^3_{i_1(l_2) i_2(l_2) i_3} \; for \; l_2 = 0, \ldots, L_1. \qquad [(5.12)](6.9)$$

Then the two-dimensional assignment problem that restores feasibility is

[(5.14)]<u>(6.10)</u>

$$\textit{Minimize} \sum_{l_2=0}^{L_2} \sum_{i_3=0}^{M_3} c^2_{l_2 i_3} z^2_{l_2 i_3}$$

$$\textit{Subject to} \sum_{i_3=0}^{M_3} z^2_{l_2 i_3} = 1, \quad l_2 = 1, \ldots, L_2,$$

$$\sum_{l_2=0}^{L_2} z^2_{l_2 i_3} = 1, \quad i_3 = 1, \ldots, M_3,$$

$$z^2_{l_2 i_3} \in \{0, 1\} \quad \textit{for all } l_2, i_3.$$

Amend the paragraph beginning at column 83, line 26, as follows:

The next objective is to remove the assumption that all cost coefficients are defined and all zero-one variables are free to be assigned. We first note that the above construction of a reduced [dimension]<u>two-dimensional</u> assignment problem ([5.13]<u>6.11</u>) is valid as long as all cost coefficients c^2 are defined and all zero-one variables in z^2 are free to be assigned. Modifications are necessary for *sparse* problems. If the cost coefficient $c^3_{i_1(l_2)i_2(l_2)i_3}$ is well-defined and the zero-one variable $z^3_{i_1(l_2)i_2(l_2)i_3}$ is free to be assigned to zero or one, then define $c^2_{l_2 i_3} = c^3_{i_1(l_2)i_2(l_2)i_3}$ as in ([5.12]<u>6.10</u>) with $z^3_{i_1(l_2)i_2(l_2)i_3}$ being free to be assigned. Otherwise, $z^2_{l_2 i_3}$ is preassigned to zero or the corresponding arc is not allowed in the feasible set of arcs. To ensure that a feasible solution exists, we now only need ensure that the variables $z^2_{l_2 0}$ are free to be assigned for $l_2 = 0, 1, \ldots, L_1$ with the corresponding cost coefficients being well-defined. [(]Recall that all variables of the form $z^3_{i_1 00}$, $z^3_{0 i_2 0}$ and $z^3_{00 i_3}$ are free (to be assigned) and all coefficients of the corresponding

form $c^3_{i_1 00}$, $c^3_{0 i_2 0}$ and $c^3_{00 i_3}$ need to be defined. [)] This is accomplished as follows: [If]if the cost coefficient $c^3_{i_1(l_2) i_2(l_2) 0}$ is well-defined and $z^3_{i_1(l_2) i_2(l_2) 0}$ is free, then define $c^2_{l_2 0} = c^3_{i_2(l_2) i_2(l_2) 0}$ with $z^2_{l_2 0}$ being free. Otherwise, since all variables of the form $z^3_{i_1 0}$ and $z^3_{0 i_2 0}$ are free to be assigned with the corresponding costs being well-defined, set $c^2_{l_2 0} = c^3_{i_1(l_2) 00} + c^3_{0 i_2(l_2) 0}$, where the first term is omitted if $i_1(l_2)=0$ and the second, if $i_2(l_2)=0$. For $i_1(l_2)=0$ and $i_2(l_2)=0$, define $c^2_{00} = c^3_{000}$.

Amend the paragraph beginning at column 83, line 52, as follows:

VI.4. The final recovery problem[.]

The recovery problem for the case [N=3]$N = 3$ is

$$\text{Minimize} \quad \sum_{l_2=0}^{L_2} \sum_{i_3=0}^{M_3} c^3_{l_2 i_3} z^2_{l_2 i_3} \qquad ([5.14]\underline{6.11})$$

$$\begin{aligned} \text{Subject to} \quad & \sum_{i_3=0}^{M_3} z^2_{l_2 i_3} = 1, \; l_2 = 1, \dots, L_2, \\ & \sum_{l_2=0}^{L_2} z^2_{l_2 i_3} = 1, \; l_3 = 1, \dots, M_3, \\ & z^2_{l_2 i_3} \in \{0,1\} \text{ for all } l_2, i_3. \end{aligned}$$

Amend the paragraph beginning at column 84, line 1, as follows:

Let Y be an optimal or feasible solution to this [2-dimensional]two-dimensional assignment problem. The recovered feasible solution z^3 is defined by

$$z^3_{i_1 i_2 i_3} = \begin{cases} 1, & \text{if } i_1 = i_1(l_2),\, i_2 = i_2(l_2) \text{ and } Y_{l_2 i_3} = 1, \\ 0, & \text{otherwise.} \end{cases} \qquad ([5.15]\underline{6.12})$$

Said in another way, let $\{(l_2(l_3), i_3(l_3)\}_{l_3=0}^{L_3}$ be an enumeration of indices $\{l_2, i_3\}$ of Y such that $Y_{l_2(l_3), i_3(l_3)} = 1$ for $(l_2(l_3), i_3(l_3)) \neq (0,0)$ and $(l_2(l_3), i_3(l_3)) = (0,0)$ for $l_3=0$ regardless of whether $Y_{00}=1$ or not. Then the recovered feasible solution can be written as

([5.16]6.13)

$$\hat{z}^3_{i_1 i_2 i_3} = \begin{cases} 1, & \text{if } i_1 = i_1(l_{12}), i_2 = i_2(l_{12}), i_3 = 1_3(l_3), \\ 0, & \text{otherwise.} \end{cases}$$

The upper bound on the feasible solution is given by

$$V_3(\hat{z}^3) = \sum_{i_1=0}^{M_1} \sum_{i_2=0}^{M_2} \sum_{i_3=0}^{M_3} c^3_{i_1 i_2 i_3} \hat{z}^3_{i_1 i_2 i_3} \qquad ([5.17]6.14)$$

and the lower by [$\Phi_3\gamma\ ^3\Delta\gamma\ ^3\Delta$] $\Phi_3(u^3)$ for any multiplier value (u^3).

Amend the paragraph beginning at column 84, line 28, as follows:

In particular, we have

$\Phi_3(u^3) \leq \Phi_3(\bar{u}^3) \leq V_3(\bar{z}^3) \leq V_3(\hat{z}^3)$, where u^3 is any multiplier value, [$\bar{u}\Phi_3\gamma\ ^3\Delta\in\ z^3$] $(\bar{u}^3)$ is a maximizer of $\Phi_3([u^3]u^3)$, $\bar{z}^3$ is an optimal solution of the multidimensional assignment problem and $\hat{z}^3$ is the recovered feasible solution defined by ([5.16]6.13).

Amend the paragraph beginning at column, 84, line 33, as follows:

VII. **Other Relaxations for the Multidimensional Assignment Problem.[(ABP)]**

In this section, we briefly describe other possible relaxations and their implications. These include linear programming relaxations and the corresponding duals.

Amend the paragraph beginning at column 84, line 38, as follows:

Recall from Section II that one starts with the definition of the zero-one variable $z_{i_N \cdots i_N} = 1$ and cost coefficients to form the problem [@]

$$\begin{aligned}
&\textit{Minimize} \sum_{i_1 \cdots i_N} c_{i_1 \cdots i_N} z_{i_1 \cdots i_N} \\
&\textit{Subject To} \sum_{i_2 i_3 \cdots i_N} z_{i_1 \cdots i_N} = 1 \qquad (i_1 = 1, \cdots, M_1), \\
&\sum_{i_1 i_3 \cdots i_N} z_{i_1 \cdots i_N} = 1 \qquad (i_2 = 1, \cdots, M_2), \qquad \underline{(7.1)} \\
&\sum_{i_1 \cdots i_{p-1} i_{p+1} \cdots i_N} z_{i_1 \cdots i_N} = 1 \\
&\qquad (i_p = 1, \cdots, M_p \ \textit{and} \ p = 2, \cdots, N-1), \\
&\sum_{i_1 i_2 \cdots i_{N-1}} z_{i_1 \cdots i_N} = 1 \qquad (i_N = 1, \cdots, M_N), \\
&z_{i_1 \cdots i_N} \in \{0, 1\} \ \textit{for all} \ i_1, \cdots, i_N,
\end{aligned}$$

where $c_{0 \cdots 0}$ is arbitrarily defined to be zero. Here, each group of sums in the constraints represents the fact that each non-dummy report occurs exactly once in a "track of data". One can modify this formulation to include *multiassignments* of one, some, or all the actual reports. The assignment problem ([2.5]3.5) is changed accordingly. For example, if $z_{i_k}^k$ is to be assigned no more than, exactly, or no less than $n_{i_k}^k$ times, then the "=1" in the constraint ([2.5]3.5) is changed to "$\leq, =, \geq n_{i_k}^k$," respectively. Modifications for group tracking and multiresolution features of the surveillance region will be addressed in future work. In making these changes, one must pay

careful attention to the independence assumptions that need not be valid in many applications.

Amend the paragraph beginning at column 84, line 54, as follows:

Again, the recent work of Poore and Drummond (A.B. Poore and O.E. Drummond. Track initiation and maintenance using multidimentional assignment problems. In D.W. Hearn, W.W. Hager, and P.M. Pardalos, editors, *Network Optimization*, volume 450, pages 407-422, Boston, 1996. Kluwer Academic Publishers B.V.) significantly extends the formulation of the track maintenance and initiation to new approaches. The discussions of this section apply equally to those formulations.

Amend the paragraph beginning at column 84, line 58, as follows:

VII.1. The Linear Programming Relaxation[.]

In the linear programming relaxation, one replaces the zero-one variables constraints

$$z_{i_1 \cdots i_N} \in \{0,1\} \text{ for all } i_1, \ldots, i_N \qquad ([6.2]\underline{7.2})$$

with the constraint

$$0 \le z_{i_1 \ldots i_N} \le 1 \text{ for all } i_1, \ldots i_N. \qquad ([6.2]\underline{7.3})$$

Then, the problem ([6.1]7.1) can be formulated as a linear programming problem with the constraint ([6.2]7.2) in ([6.1]7.1) replaced by ([6.3]7.3) with a special block structure to which the Dantzing-Wolfe decomposition is applicable. Of course, after solving this problem, one must now recover the zero-one character of the variables in ([6.1]7.1) and there are many ways to do this, such as using the [two dimensional] two-dimensional assignment problems. Commercial software is also available.

Amend the paragraph beginning at column 85, line 7, as follows:

VII.2. The Linear Programming Dual and Partial Lagrangian Relaxations[.]

Given the linear programming relaxation, one can formulate the dual problem or the partial Lagrangian relaxation duals with respect to any number of constraints. In particular, this is precisely what is done in Section 3 on the Lagrangian relaxation algorithm presented. The much broader class of algorithms provided in the US Patent 5537119 (Aubrey B. Poore, Jr., Method and System for Tracking Multiple Regional Objects by [Multi Dimensional] Multi--Dimensional Relaxation, US Patent Number 5537119, filed 14 March 1995, issue date of 16 July 1996) can be modified to remove the zero-one character when one relaxes *M* sets of constraints to an *(N-M[dimensional])*-dimensional problem and recovers with an [N-M÷1 dimensional] *(N-M* +1)-dimensional problem. This avoids the problems associated with the NP-hard *(N-M)*- and [N-M+1] *(N-M* +1)- dimensional problems. However, to restore the zero-one character, one can do it sequentially with an assignment problem or with one of the many zero-one rounding techniques. These formulations are easy to work out and thus the details are omitted.

Delete the paragraphs beginning at column 85, line 31, thru column 85, line 39.

Amend the paragraph beginning at column 85, line 40, as follows:

VIII. Hot Starts for Track Maintenance and Initiation: Bundle Modifications [(ABP)

Amend the paragraph beginning at column 85, line 43, as follows:

Thus reuse of multipliers and the first proof that this reuse is actually valid was presented in this section. The reuse in track maintenance is demonstrated and discussed in the work of Poore and Drummond [[xx]] (A.B. Poore and O.E. Drummond. Track initiation and maintenance using multidimensional assignment problems. In D.W. Hearn, W.W. Hager, and P.M. Pardalos, editors, *Network Optimization*, volume 450, pages 407-422, Boston, 1996. Kluwer Academic Publishers B.V.). The only item left is the modification of the bundle of subgradients for the use with the multiplier values as one goes through the recovery problem as in [Section 3.6] Section IV.3.6 or as one moves the window in track maintenance. It is this aspect of the nonsmooth optimization that adds an order of magnitude to the speed of the algorithms. This work is based on both a mathematical proof as in [Section 3.6] Section IV.3.6 as well as computational experience and heuristics.

Amend the paragraph beginning at column 85, line 55, as follows:

IX. [**Adaptive Window Size Selection (ABP).**] ADAPTIVE AND VARIABLE WINDOW SIZE SELECTION

These data association algorithms, based on the multidimensional assignment problem, range from sequential processing to many frames of data processed all at once! The data association problem for [multisensor multitarget] multisensor-multitarget tracking is formulated using a window sliding over the frames of data from the sensors. This discussion focuses on the work of Poore and Drummond and not on

the formulations in [Section 2]Section 2 III. Firm data association decisions for existing track are made at, say frame *k*, with the most recent frame being *k+M*. Decisions after frame *k* are soft decisions. Reports not in the confirmed tracks are used to initiate tracks over frames numbered *k-I* to *k+M*.

Amend the paragraph beginning at column 86, line 1, as follows:

The length of these windows varies from sequential processing, which corresponds precisely to [I=0]$I = 0$ and [M=1]$M = 1$, to user defined large values of *I* and *M*.[. (In the case of sequential processing, we also have a temporary track file of dynamically feasible tracks, but incorrect data association.)] There is a marked change in performance over this range. As the two windows become exceedingly long, there is little statistically significant information gained and indeed performance can degenerate slightly. Thus, one must manually find the correct window lengths for performance in a given scenario and the algorithms do not change for a changing environment. Thus we propose an *adaptive* method for adjusting the window lengths. (The method has been highly successful for selecting the correct window length for multistep methods in ordinary differential equations.)

1. Compute the solution for different window lengths that overlap and differ by one or two frames.

2. Compare the solution quality, i.e., the value of the objective function, for two adjacent window lengths.

3. If there is a predefined improvement in either direction, we then, for stability, repeat the exercise for a shorter

or longer on the first try. If there is consistent information, we adjust the window size in the indicated direction. This can be given a well defined mathematical formula in terms of the assignment problems of different dimensions.

[• Algorithm Enhancements due to Data Structures (ABP).
- Construction and enumeration of the Best K-Solutions.
- New Nonsmooth Optimization Techniques.
- Sensitivity Analysis]

Amend the paragraph beginning at column 86, line 32, as follows:

X. SENSITIVITY ANALYSIS

For sequential processing in which the [two dimensional] two-dimensional assignment algorithm is used, one can use the LP sensitivity analysis theory and easily obtain the corresponding answers. For the multiframe processing, the optimal solution is not available; however, there are still two approaches one can use. First, the basic algorithm can perform the same sensitivity analysis at each stage (loop) of the algorithm as is done in the [two dimensional] two-dimensional assignment problem since the evaluation of function Φ_{N-k+1} is equivalent to a [two dimensional] two-dimensional assignment problem. Alternately, one could use an LP relaxation of the assignment problem and base the sensitivity on the resulting LP problem. We currently see this as an important step in finding even better solutions to the assignment problem if so desired.

Amend the paragraph beginning at column 86, line 46, as follows:

XI. [New Auction Algorithms (ABP & PS)]NEW AUCTION ALGORITHMS.

In this section we present a new auction algorithm for the [two dimensional] two-dimensional assignment problem.

Amend the paragraph beginning at column 86, line 49, as follows:

An important step in solving N-dimensional assignment problems for $N \geq 3$[,] is finding the optimal solution of the [2-dimensional]two-dimensional assignment problem. In particular, we wish to solve the [2-dimensional]two-dimensional problem which includes the zero-index. [3,]Typically this problem can be thought of as trying to find either a minimum cost or maximum utility in assigning person to objects, tenants to apartments, or teachers to classrooms. We will follow the work of Bertsekas and call the first index set persons and the second index set objects. The statement of this problem is given below (11.1) when the number of persons is m and the number of objects is n.

[(1)](11.1)

$$\textit{Minimize} \quad \sum_{i=0}^{m}\sum_{j=0}^{n} c_{ij}x_{ij}$$

$$\textit{Subject to} \quad \sum_{j=0}^{n} x_{ij} = 1 \quad \textit{for all } i = 1,\ldots,m,$$

$$\sum_{i=0}^{m} x_{ij} = 1 \quad \textit{for all } j = 1,\ldots,n.$$

Amend the paragraph beginning at column 87, line 7, as follows:

There are a couple of assumptions which we will make about [(1)](11.1). First of all, we assume that c_{i0} and c_{0j} are well-defined and the corresponding variables [x_{i0}]$\underline{x}_{i0}$ and [x_{0j}]$\underline{x}_{0j}$ are

free to be assigned. Second if a cost c_{ij} happens to be undefined, then the corresponding variable [x_{ij}]$\underline{x_{ij}}$ will be set to 0. In effect if c_{ij} is undefined, we simply remove this cost and variable from inclusion in the problem.

Amend the paragraph beginning at column 87, line 15, as follows:

Notice that there are no constraints on the number of times person 0 and object 0 can be assigned. But notice that the first constraint requires that each non-zero person i must be assigned exactly once. Similarly, the second constraint forces each non-zero object j to be assigned exactly one time. Finally, the last constraint gives an integer solution, although we will see shortly that this constraint can be relaxed to admit any solution [$x_{ij} \geq 0$]$\underline{x_{ij} \geq 0}$. One reason for requiring that all of the costs of the form [c_{i0}]$\underline{c_{i0}}$ and [c_{0j}]$\underline{c_{0j}}$ be defined is so that we are guaranteed a feasible solution exists for the given problem.

Amend the paragraph beginning at column, 87, line 25, as follows:

XI.1. Relaxation of One Constraint[.]

Relaxing the last constraint, we obtain the following problem:

$$\begin{aligned} \textit{Minimize} \quad & \sum_{i=0}^{m} \sum_{j=0}^{n} c_{ij} x_{ij} + \sum_{j=0}^{n} u_j^2 \left[\sum_{i=0}^{m} x_{ij} - 1 \right] \\ & \equiv \sum_{i=0}^{m} \sum_{j=0}^{n} \left[c_{ij} + u_j^2 \right] x_{ij} - \sum_{j=0}^{n} u_j^2 \\ \textit{Subject to} \quad & \sum_{j=0}^{n} x_{ij} = 1 \quad \textit{for all } i = 1, \ldots, m \\ & x_{ij} \in \{0, 1\}. \end{aligned}$$

Amend the paragraph beginning at column 87, line 57, as follows:

XI.2. Relaxation of Both Constraints[.]

This time we will relax both constraints and [using]use duality theory to obtain a relationship between the multipliers u^1 and u^2.

Delete the paragraphs beginning at column 87, line 62, thru column 87, line 65.

Amend the paragraph beginning at column 88, line 39, as follows:

In the selection of i_j[.] above, if a tie occurs between 0 and any non-zero index k, then select j_i as k. Otherwise, if there is a tie between two or more non-zero indices, the choice of j_i is arbitrary.

Also if $B(j)$ consists of only one element, then set $\gamma_j = \infty$.

Delete the paragraph beginning at column 89, line 20, thru column 89, line 22.

Insert the following heading before the paragraph beginning at column 89, line 23:

XII. SOME CONCLUDING COMMENTS

Amend the paragraph beginning at column 89, line 26, as follows:

Step 2 is the computational part of the algorithm. Evaluating Φ_{N-k+1} [XXX] and computing a subgradient use in the search procedure requires 99% of the computing time in the algorithm. This part uses [two dimensional] two-dimensional assignment algorithms, a search over a large number of indices, and a nonsmooth optimization algorithm. It is the

second part (the search) that consumes 99% of the computational time and this is almost entirely parallelizable. Indeed, there are [two dimensional] two-dimensional assignment solvers that are highly parallelizable. Thus, we need but parallelize the nonsmooth optimization solver to have a reasonably complete parallelization.